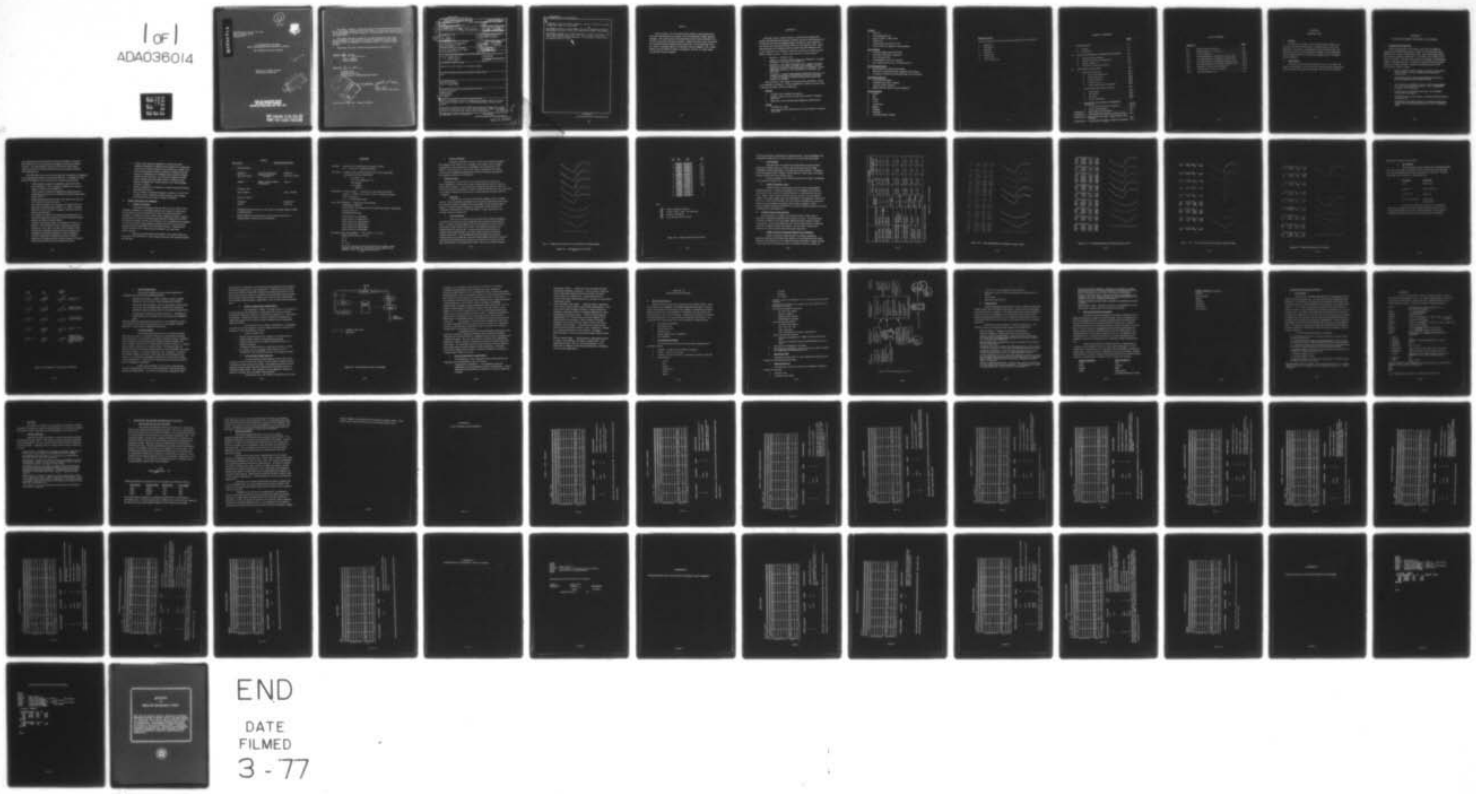
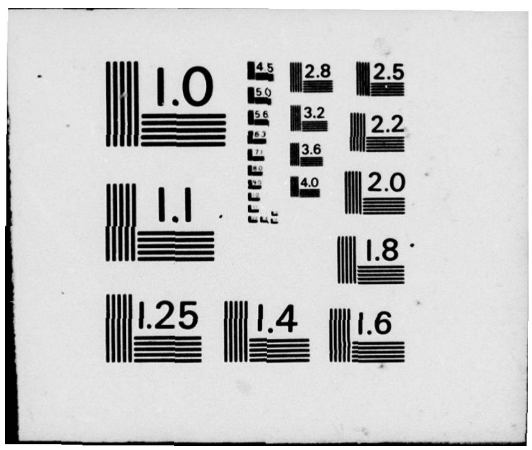
AD-A036 014 PRC INFORMATION SCIENCES CO MCLEAN VA F/G 8/2
ACS SYMBOLIZATION FOR DMAAC. VOLUME I. SYSTEM IMPLEMENTATION AN--ECT(U)
NOV 76 P D BELL, J A NEUFFER, M L TAYLOR F30602-75-C-0319
UNCLASSIFIED RADC-TR-76-334-VOL-1 NL

1 OF 1
ADA036014



END
DATE
FILMED
3-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6

(12)
NW

RADC-TR-76-334, Volume I (of two)
Final Technical Report
November 1976

ACS SYMBOLIZATION FOR DMAAC
System Implementation and Operating Procedures

PRC Information Sciences Company

2 A035993

Approved for public release;
distribution unlimited.

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report contains a large percentage of machine-produced copy which is not of the highest printing quality but because of economical consideration, it was determined in the best interest of the government that they be used in this publication.

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

This report has been reviewed and approved for publication.

APPROVED:

JOHN R. BAUMANN
Project Engineer

APPROVED:

HOWARD DAVIS
Technical Director
Intelligence & Reconnaissance Division

FOR THE COMMANDER:

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 **REPORT DOCUMENTATION PAGE**

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 18 RADC-TR-76-334 - Vol-1 (of two)	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER: 9	
4. TITLE (and Subtitle): 6 ACS SYMBOLIZATION FOR DMAAC. Volume I. System Implementation and Operating Procedures.	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report. June 1975 — April 1976.
	6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): 10 Paul D./Bell, John A./Neuffer, M. Lynn/Taylor	8. CONTRACT OR GRANT NUMBER(s): 15 F30602-75-C-0319
9. PERFORMING ORGANIZATION NAME AND ADDRESS: PRC Information Sciences Company, 7600 Old Springhouse Road, McLean VA 22101	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 64701B, 16 32020316, 17 03
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (IRRP), Griffiss AFB NY 13441	12. REPORT DATE: 11 November 1976
	13. NUMBER OF PAGES: 60
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same 12 69p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer:
John R. Baumann (IRRP)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Digital Cartography
Symbolization
Line Smoothing
Cartography

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

RADC has implemented a Graphic Line Symbolization System (GLSS) on the Univac 1108 computer system located at the Defense Mapping Agency Aerospace Center (DMAAC).

The software accepts data in the DMAAC Lineal Input System format and creates symbolized line data for final color separation plotting. Symbols are applied according to Joint Operations Graphics (JOG) specifications. (see reverse) over

407126 LB

In addition to the lineal symbol capability, the GLSS is capable of creating a significant number of point symbols.

5

The software is written in ASCII COBOL and Fortran V languages and requires approximately 40K words of memory for loading and execution of all functions.

The software configuration is highly segmented into areas of job set-up, file input, job monitoring, symbol application control, symbol specification correlation, symbol application processes, line smoothing and data culling, job reporting, and file output.

PREFACE

This is Volume I of a two volume final technical report prepared by PRC Information Sciences Company, 7600 Old Springhouse Road, McLean Virginia. The report covers work performed under Contract F30602-75-C-0319 for Rome Air Development Center, Griffiss Air Force Base, New York. The report describes work performed from June 1975 through April 1976. Mr. John R. Baumann (IRRC) was the RADC Project Engineer, Mr. Frank Mirkay was the DMAAC technical coordinator, and Mr. M. Lynn Taylor was the PRC Project Manager.

ABSTRACT

PRC/ISC, under a RADC contract, converted and expanded the Graphic Line Symbolization System (GLSS) to operate in the production environment at the Defense Mapping Agency Aerospace Center (DMAAC). The system was converted from the HIS-635 to the UNIVAC 1108. Major enhancements to the original software system included additional point symbol capabilities and additional output file formats for interfacing with DMAAC plotter systems. Major characteristics of GLSS includes:

- o Hardware - UNIVAC 1108
- o Software - written in ASCII COBOL and FORTRAN V compiler languages and operates under EXEC 8.
- o Modularity - the software configuration is highly segmented into areas of job setup, file input, job monitoring, symbol application control, symbol specification correlation, symbol application processes, line smoothing, job reporting, and file output.
- o Resources - requires approximately 40K words of memory for loading and execution of all software; program overlaying or selective loading of required software can significantly reduce core storage.

GLSS provides a wide range of data processing capabilities related to cartographic symbology. Various capabilities, options and processing techniques of GLSS includes the following:

<u>Dashing</u>

- o Variable sizes of dashes and spaces
- o Feature must start and end with dash at least 1/2 length of dash size
- o Dash must carry through points flagged as special points

<u>Casing</u>

- o Variable size cases
- o Line quality of case should equal or exceed quality of original line center

Ticking

- o Full tick
- o Alternating half tick
- o Half tick (left or right code)
- o Double tick
- o Feature will not end with a tick
- o Ticks will not be applied at special points

Line Cleaning

- o Cleaning angles (angle bisecting)
- o Minimum resolution maintenance
- o Line "back-up" edit
- o Combinations of the above options
- o Data culling based on line inclination factors

Symbol Specifications

- o Specification file building and update
- o Selection of specification file (multiple product files)
- o Override to standard specifications (up to 10 overrides)

Input/Output Options

- o LIS Table File (Input)
- o GERBER 2032 Plotter (Output)
- o Xynetics Plotter (Output)
- o MMS-32/Raster Plotter Interface (Output)

Point Symbology

- o Circle
- o Dot
- o Arrow
- o Cross
- o Half-Arrow
- o Square
- o Triangle
- o Pyramid
- o Arc/Chord (Mine Symbol)

Multiple Symbols

Various combinations of point and lineal symbology can be generated.

- o Dash/Case
- o Dash/Cross
- o Dash/Dot
- o Dash/Tick
- o Dash/Circle
- o Line/Arrow
- o Line Center/Case

TABLE OF CONTENTS

LIST OF FIGURES

SECTION I

INTRODUCTION

A. Purpose

The purpose of Volume I of the Final Technical Report is to: present an overview of the project, including system conversion, expansion, test results, and potential system improvements; and define the procedures for operating the Graphic Line Symbolization System (GLSS) installed at the Defense Mapping Agency Aerospace Center (DMAAC).

B. Organization

Section II of this volume describes the system conversion, expansion, and testing. Section III of this volume presents a description of the operating concepts and procedures and control card formats.

SECTION II

SYSTEM CONVERSION, EXPANSION, AND TESTING

A. Background and Summary

PRC Information Sciences Company previously developed an experimental symbolization system at Rome Air Development Center (RADC) under Contract F30602-74-C-0027. The experimental system was written in COBOL, FORTRAN Y, and GMAP, and operates on the HIS-635 at RADC. Basically, the system was directed at restructuring digital cartographic features to conform to standardized lineal symbol patterns for digital plotting. Design and development goals of the experimental system included:

- system should be flexible enough to support a wide range of product requirements and expandable to support new requirements;
- job execution should require approximately 32K words of memory (maximum 60K words of memory);
- user control for variability of symbol application in support of a variety of production processes, e.g., compilation, proofing, and final plotting;
- improvements in processing efficiencies over previous symbolization algorithms;
- improvements in product quality, carefully balancing efficiency and quality; and
- compatibility with DMAAC UNIVAC 1108 programming standards (reference Operating Instruction 171-23, 12 February 1972).

The experimental system provided for input and output of a standard RADC MMS-32 file for interfacing with digitizer and plotter systems at RADC. The symbolization repertoire included all major lineal symbols plus the set of point symbols which are normally associated with lineal patterns.

PRC/ISC under this contract was tasked with conversion and expansion of the experimental software to operate at DMAAC. A summary of work performed during the course of the project included the following:

- GLSS computer programs, originally written GE COBOL and FORTRAN V, were compiled, debugged, and tested on DMAAC's UNIVAC 1108 (System A).
- A new software routine was designed and implemented for producing plot tapes for the GERBER 2032 Plotter (photo head Model OEH-B).
- Symbol specification library files were defined and built for ATC 200 and JOG Series Charts.
- New software routines were designed and implemented for generating additional point symbols (i.e. square, triangle, pyramid, and arc/chord) in compliance with ATC and JOG product specifications.
- New software routines were designed and implemented for formatting output files to interface with the Xynetics Plotter System and MMS-32/RAPS software.
- LIS input processing routines were upgraded to operate at DMAAC, with provisions for processing feature headers through the class, type, sub-type, and descriptor levels.
- Analysis and preliminary design work was performed for developing an expanded double-line symbol capability.
- A feature suppression capability was added to GLSS to symbolize and output only those feature groups which match feature groups included in the specifications files.

- A feature code masking capability was added to provide additional flexibility in defining feature groups to be symbolized.
- A stand alone software program was installed for reading a LIS file and generating a summary of features residing on the file.
- Software modifications were installed which resulted in major symbol improvements for: (1) producing continuous symbol pattern through data point buffers; and (2) adding new tick symbol capabilities for "double half-tick" and "dash/multiple half-tick" symbols.
- Software programs were delivered to DMAAC along with interim operating procedures.
- Over 3-months of resident maintenance services were provided to DMAAC which consisted of system testing, user training, and detection and correction of software problems.

B. System Conversion and Expansion

1. System Conversion

Conversion of programs from the HIS-635 to the UNIVAC 1108 basically consisted of: performing required code changes to source files on the HIS-635 to allow for differences in the Honeywell and UNIVAC compilers; creating card decks of source files required at DMAAC; loading each program as symbolic elements under disk file DBM*UCPR-LT using the data base manager system on the UNIVAC 1108; compiling and de-bugging each source program and preparing relocatable elements; and testing the cycling and basic capabilities of the converted software. FORTRAN and COBOL versions of the GLSS common data buffer were built using Blank Common.

Differences in HIS COBOL and UNIVAC 1108 ASCII COBOL and FORTRAN Y and FORTRAN V which required program code changes included the following:

COBOL

HIS COBOL		UNIVAC ASCII COBOL
o Word Formats:		
DISPLAY	6 Bit BDC/Field Data	DISPLAY-1
PIC 9(8) Comp-1	Fixed Point Binary (36 Bits)	PIC 9(10) Comp-4
Comp-2	Single precision signed floating point	Comp-1
o Program Calls:		
CALL NAME		CALL "NAME"
o Reserved Words:		
MESSAGE		MESSAGEU
COUNT		COUNTZ

o Working storage section must come prior to common storage section in ASCII COBOL.

o ASCII COBOL does not generate a dummy word at the start of common (this is contrary to documentation).

FORTRAN

Indexing - Cannot use a variable index to acquire an index

i.e. - IXYZ (1, I CORDX (ICURDX))

DO Loops - Cannot use the variable of the DO for any set value after leaving the DO loop (FORTRAN)

i.e. -
```
DO 100 I=1, N
If (I .EQ. INDX) go to 110
100 continue
110 continue
INONO = I
```

Subroutine as a main Routine - cannot have a subroutine as the main routine. An IGDM error (Guard mode or undefined sequence fault) is produced.

i.e. - SUBROUTINE MAIN

DATA STATEMENTS - Cannot have the following:

```
DIMENSION ITEXT(7)
DATA ITEXT /37H THIS IS NOT ALLOWED IN UNIVAC FORTRAN/
```

It can be done as:

```
DIMENSION ITEXT(6)

DATA ITEXT (1) /6HTHISbI/
DATA ITEXT (2) /6HSbALLO/
DATA ITEXT (3) /6HWEDbBY/
DATA ITEXT (4) /6HbUNIVA/
DATA ITEXT (5) /6HCbFORT/
DATA ITEXT (6) /6HRANbbb/
```

Subroutine CALL STATEMENT - CALL SUB (1, 12, 65.4)

```
SUBROUTINE SUB (I1, I2, R1)
I1 = 4
I2 = 0
R1 = 6.12
```

Note: the above may cause the location in the calling routine (locations containing zero twelve and six five point 4) to be changed to four, zero and six point one two.

2. System Expansion

Expansion and improvements to GLSS were primarily designed and installed on the HIS-635 for compiling, unit testing, and de-bugging. Installation and full testing of new capabilities were performed on the UNIVAC 1108 (System A). Most of the software integration and testing occurred on an individual change basis to allow for modular expansion of the system and interim use by DMAAC.

C. System Testing

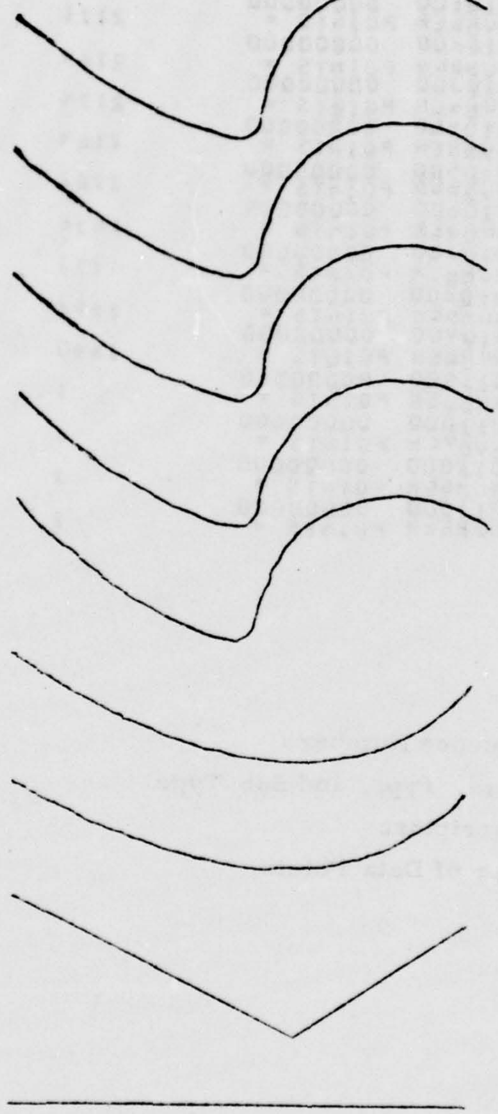
The purpose of this section is to describe results of testing the basic capabilities of GLSS; and in doing so illustrate the variety of symbols which can be produced. The testing also reveals process timings which can be used for run time projections. DMAAC conducted similar and more voluminous tests which contributed to verification of capabilities and detection of error conditions.

1. Test Data

A test file was created on the LIS and output in table coordinate format. The test graphic was digitized at 20 micron setting (40 micron steps) and consisted of features digitized via single point, trace, and point-point modes. To minimize processing time for testing, the file consisted of 13 features represented by 17,855 data points. A line center plot of the test file is shown in Figure II-1.

2. Test Procedures

Each test was conducted to verify the basic symbol capabilities of GLSS and to identify process timings for major types of symbol applications. A header summary (Figure II-2) of all features on the test file was first run to verify feature header codes. The purpose of each test was defined followed by job set-up. Set-up normally included creation of specific symbol specification cards which would exercise the desired software processes. The specification override option was normally employed since our testing was directed at illustrating the symbol repertoire. Following job execution the job report was reviewed for accuracy and the plot was prepared on the appropriate plotter. Plots were closely examined



Note - 4 single point features are not viewable on Xynetics plots.

Figure II-1 Xynetics Plot of Test File

```
 1     2          3                 4

 1   010100   00000000
     NUMBER   POINTS =           2171
 2   010200   00000000
     NUMBER   POINTS =           2169
 3   010300   00000000
     NUMBER   POINTS =           2174
 4   010400   00000000
     NUMBER   POINTS =           2169
 5   010500   00000000
     NUMBER   POINTS =           2182
 6   010600   00000000
     NUMBER   POINTS =           1836
 7   010700   00000000
     NUMBER   POINTS =           1773
 8   010800   00000000
     NUMBER   POINTS =           1692
 9   010900   00000000
     NUMBER   POINTS =           1690
10   011000   00000000
     NUMBER   POINTS =              1
11   011000   00000000
     NUMBER   POINTS =              1
12   011000   00000000
     NUMBER   POINTS =              1
13   011000   00000000
     NUMBER   POINTS =              1
```

Note:

1 Feature Sequence Number
2 Feature Class, Type, and Sub-Type
3 Feature Descriptors
4 Total Number of Data Points

Figure II-2 Header Summary of Test File

to verify that proper symbology was being generated. Process timings were computed for both SUP and CPU and based on seconds/1000 data points.

3. Run Timings

Execution time for GLSS runs is dependent on several factors primarily including number of feature data points processed by input, smoothing, symbol application, and output processing routines. Variations in processing parameter settings, smooth options, symbol types, and output formats further affect the execution time.

Based on sample runs conducted against the test file, run timings are presented in Figure II-3.

4. Symbol Repertoire Tests

GLSS generates cartographic symbols based on specifications defined by the user. Symbol specifications can be supplied to the system via standard files created for specific products or provided in the job card deck. While a wide variety of symbol patterns and sizes can be produced a finite set of legal symbol types and combinations are available. Figures II-4, II-5, II-6, and II-7 demonstrates the basic symbol definitions and associated patterns for conventional symbology which GLSS can support. Other non-conventional symbol combinations are possible although should be tested and verified prior to production use.

D. Potential System Improvements

The GLSS system at DMAAC currently provides a wide range of automated symbology capabilities and related processes. Processing time for GLSS execution is very reasonable considering data volume and types of processes. Several potential areas for system improvement do exist and should be considered by the Government. Intersections of double-line symbols is a major problem and is discussed separately.

1. Design Concept for Improved Double-Line Symbology

RADC and DMAAC requested PRC to examine symbology problems associated with double-line symbols currently generated by GLSS. The goal of the analysis was to scope the problem and define a design concept which could lead to development of new or modified software routines for

RUN TYPE	OUTPUT FORMAT	SMOOTH OPTION	RUN TIMINGS Total Seconds SUP	CPU	Seconds/1000 Data Points Input* SUP	CPU
1. Line Center	Xynetics	0	26.339	18.759	1.5	1.1
2. Line Center	GERBER (no print-out)	0	76.088	63.097	4.3	3.5
3. Line Center	MMS-32	0	15.101	5.401	.8	.3
4. Line Center & Smoothing	Xynetics	4 (Min. Dist. .003")	27.315	20.499	1.5	1.1
5. Dash Symbology	Xynetics	0	24.466	17.027	1.4	1.0
6. Dash Symbology	GERGER (No Printout)	4 (Min Dist. 004")	32.882	24.669	1.8	1.4
7. Tick Symbology	Xynetics	0	30.569	22.641	1.7	1.3
8. Tick Symbology	Xynetics	4 (Min Dist .006")	16.896	10.182	.9	.6
9. Double Line Symbology	Xynetics	0 (Min Dist .006")	34.004	26.671	1.9	1.5
10. Double Line Symbology	Xynetics	2 (Min Dist .004")	23.320	16.281	1.3	1.0

* Input File Contained 17,855 Data Points.

Figure II-3 - Test Run Timings

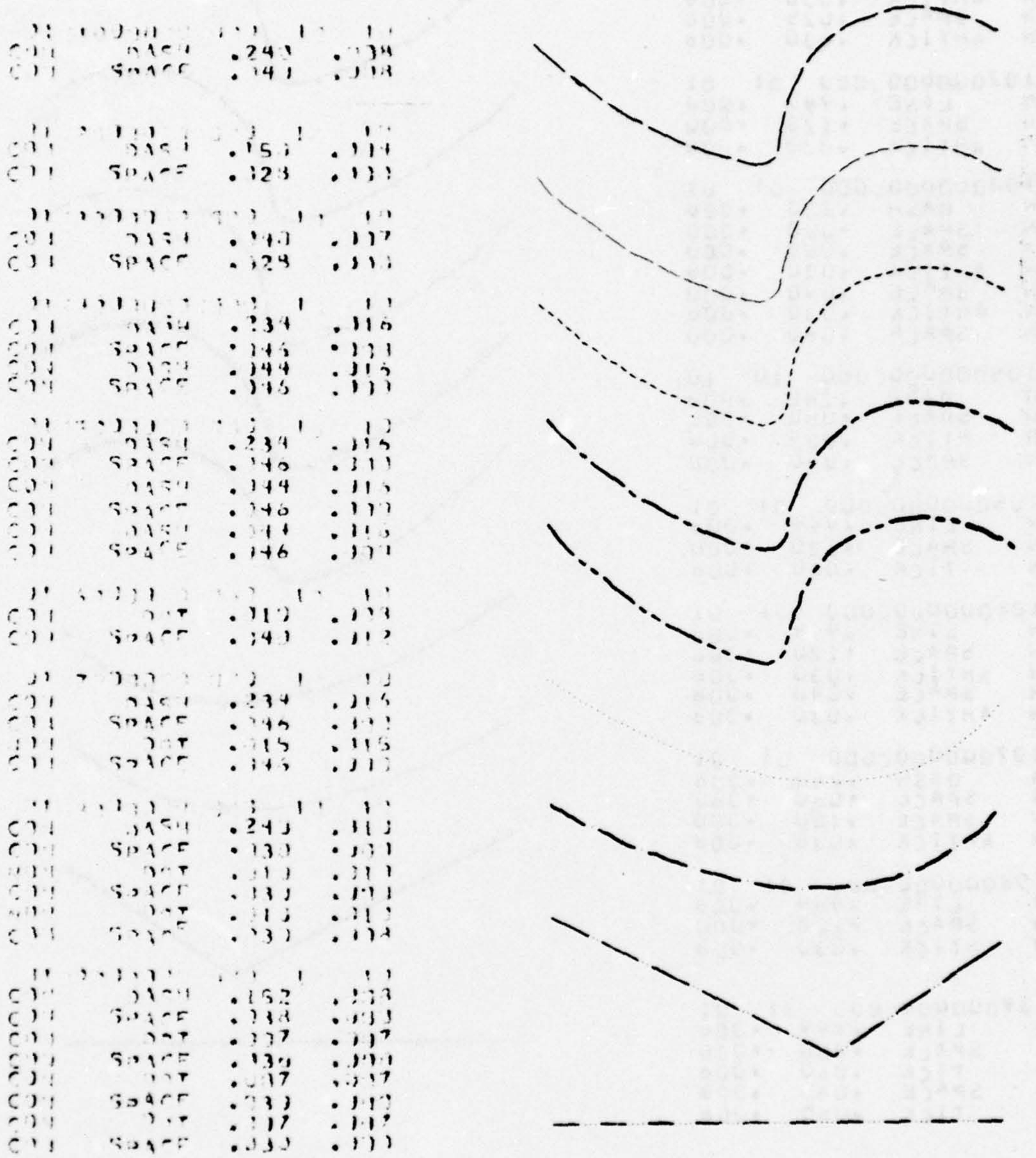


Figure II-4 Dash Specifications and Symbols (Gerber Plot)

```
0101000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .120  .000
CON    AHTICK   .030  .008
CON     SPACE   .025  .000
CON    AHTICK   .030  .008

0102000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .120  .000
CON    AHTICK   .030  .008

0103000000000   01   01
CON      DASH   .200  .008
CON     SPACE   .050  .000
CON     SPACE   .060  .000
CON    AHTICK   .030  .008
CON     SPACE   .040  .000
CON    AHTICK   .030  .008
CON     SPACE   .080  .000

0104000000000   10   10
CON      DASH   .240  .008
CON     SPACE   .050  .000
CON     HTICK   .030  .008
CON     SPACE   .040  .000

0105000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .120  .000
CON      TICK   .060  .008

0106000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .120  .000
CON    AHTICK   .030  .008
CON     SPACE   .040  .008
CON    AHTICK   .030  .008

0107000000000   01   01
CON      DASH   .240  .008
CON     SPACE   .050  .000
CON     SPACE   .120  .000
CON    AHTICK   .030  .008

0108000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .120  .000
CON     HTICK   .030  .008

0109000000000   01   01
CON      LINE   .999  .008
CON     SPACE   .250  .000
CON      TICK   .060  .006
CON     SPACE   .040  .008
CON      TICK   .060  .006
```

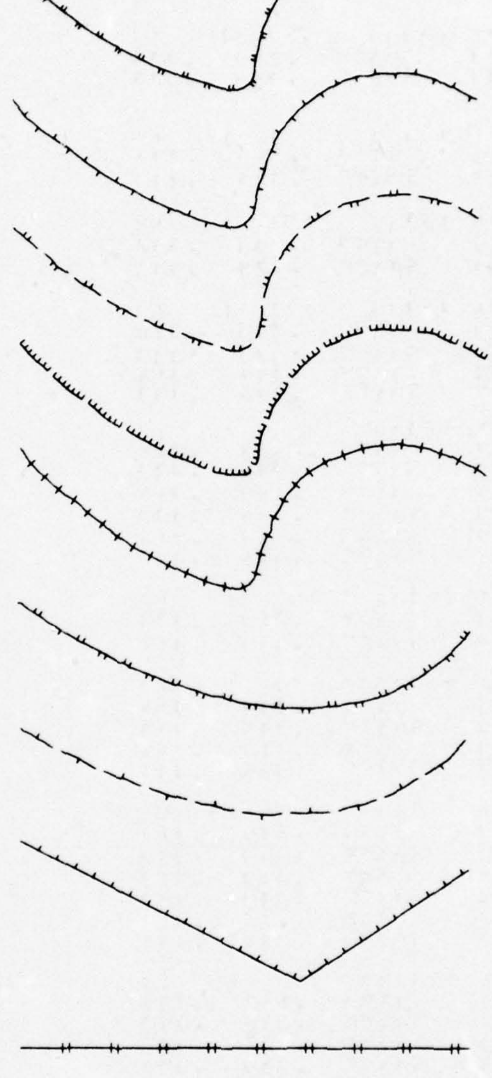


Figure II-5 Tick Specifications and Symbols (Gerber Plot)

```
01 1000000 000   10   10
CON      CASE   .020  .008

01 2000000 000   10   10
CON      CASE   .026  .008

01 3000000 000   10   10
CON      CASE   .032  .008

01 4000000 000   10   10
CON      CASE   .040  .008

01 5000000 000   10   10
CON      CASE   .036  .010

01 6000000 000   10   10
CON      DASH   .100  .008
CON     SPACE   .025  .008
CON      CASE   .028  .008

01 7000000 000   10   10
CON      DASH   .160  .008
CON     SPACE   .060  .000
CON      CASE   .026  .008

01 8000000 000   10   10
CON      LINE   .999  .007
CON      CASE   .044  .008

01 9000000 000   10   10
CON      CASE   .024  .008
```

Figure II-6 Case Specifications and Symbols (Xynetics Plot)

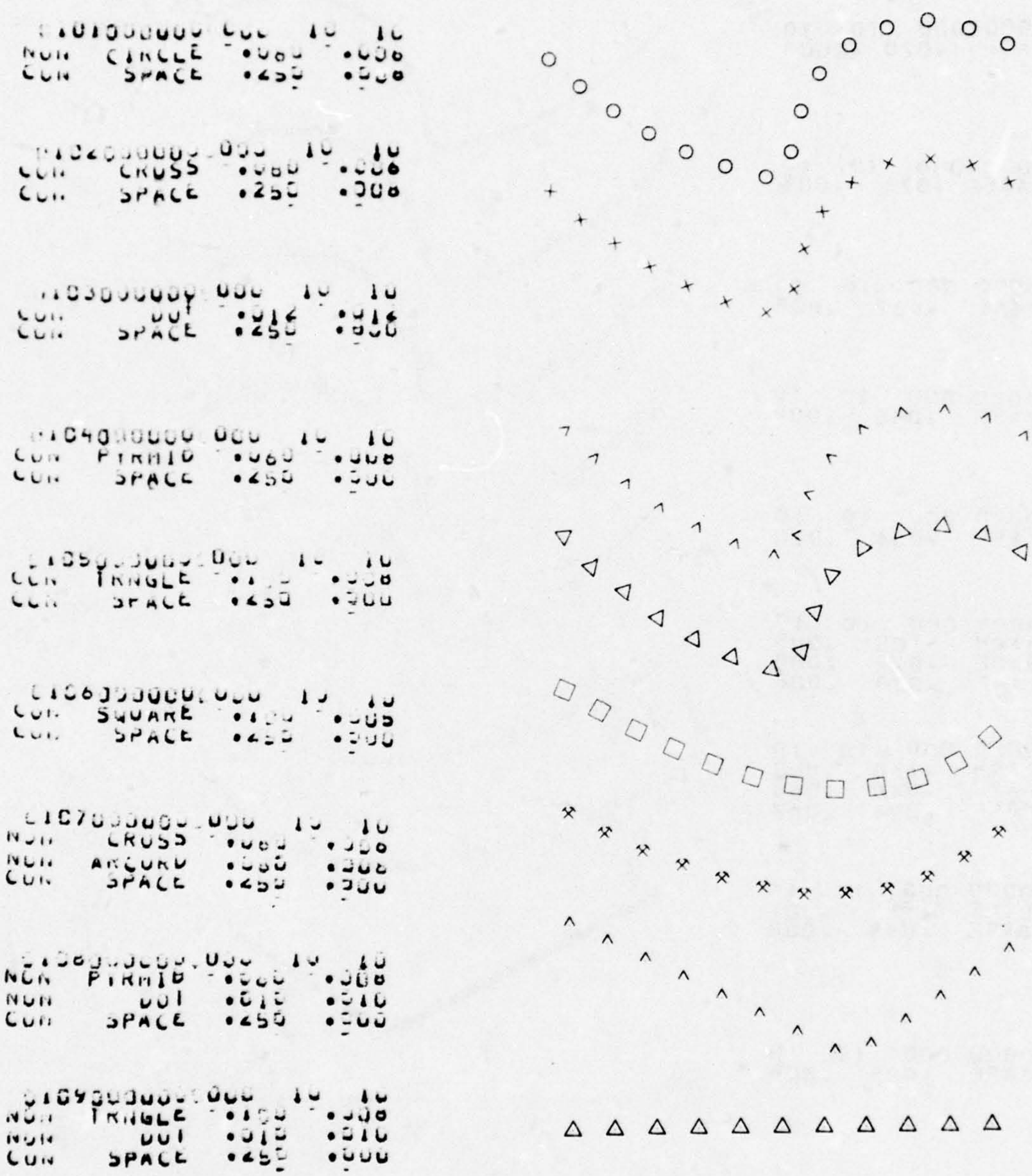


Figure II-7 Point Specifications and Symbols

producing an improved end product.

a. <u>The Problem</u>

GLSS currently produces double-line symbology directly from line-center data residing in a common buffer area; and generates case symbols independent of other feature lines or symbols. Current symbology directly related to casing includes:

SYMBOLS	FEATURE
Case	Major Road
Line/Case	Dual Lane Road
Dash/Case	Road U/C
Arrow/Case/Arrow	Bridge with Major Road

Since the generated case symbols are completely independent of other features and symbols being generated, several undesirable situations can result. Illustrated below are examples of conditions which can result from current double-line symbology.

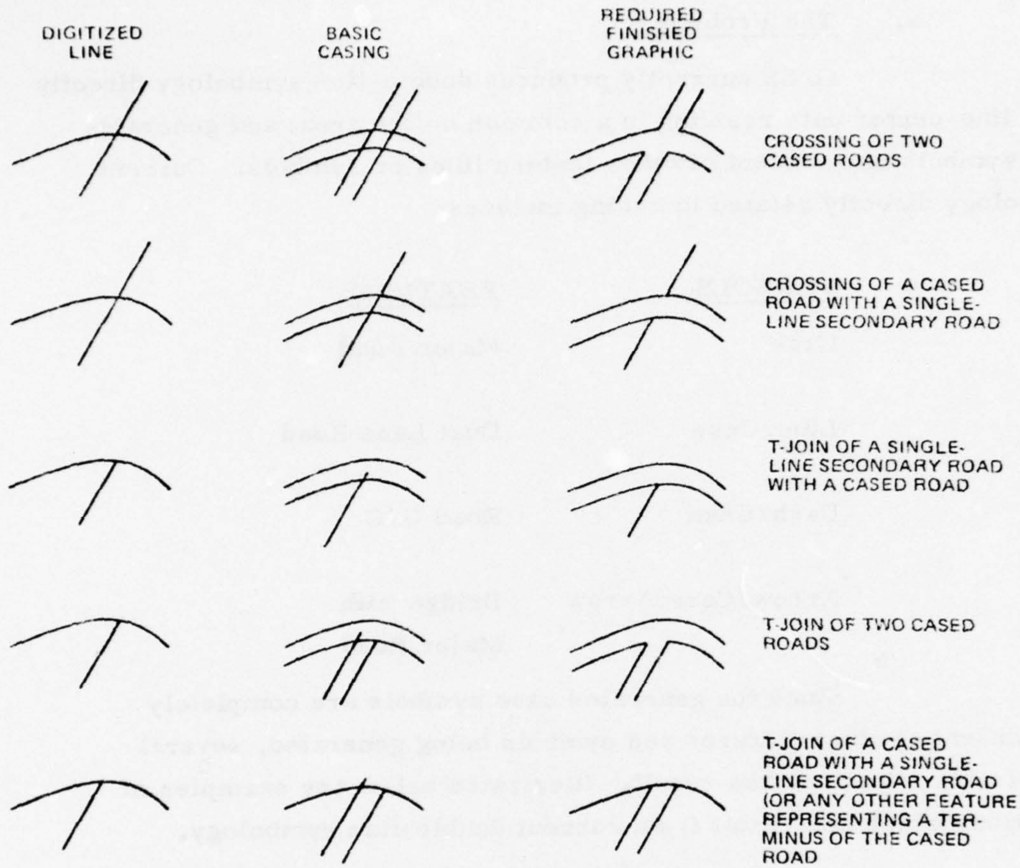


Figure II-8 Examples of Double-Line Symbology

b. General Approaches

Three general approaches have been considered for resolution of the double-line symbology problem.

- o Software Processing - software routines to detect conflicts at intersections and clip or join the appropriate symbols.
- o Interactive Symbol Editing - provide edit software which would rely on the user to locate feature symbols to be edited and identify the type of software edit to be automatically performed.
- o Photographic Masking and Manual Touch-Up - masking of interior portions of cased lines and manual touch-up of finished plots.

While the latter two approaches have merit and may eventually be used to some extent, the direction of this analysis was to investigate "software processes " which could be applied to the problem.

c. Problem Analysis

Three information items must be determined such that software processes can accurately clean-up case intersection problems. These items are: (1) detection and holding of symbols which are candidates for edit actions; (2) determining which candidate symbols conflict with other candidate symbols; and (3) determining what type of edit (i.e., segment join or clip) is required and at what locations. The types of edits are affected by size of double-line symbols generated and orientation of two lines forming a junction. Two major approaches were examined: (1) perform intersection editing simultaneous with double-line symbolization of each feature or; (2) completely symbolize the file prior to intersection editing. A major question is -- is intersection editing predictive enough to perform prior to building all symbology?

Double-line symbology creates twice as many feature symbols and approximately twice as many data points which would require storage and manipulation. The process is predictive, although the time

which would be consumed to accurately derive the prediction would probably overwhelm any gains due to data compactness. Thus the approach decided upon was to build all double-line symbols as currently performed and store all generated symbols which are candidates for intersection edit processing. This approach would yield the cleanest end product and have least impact, and associated implementation complexities, on the current GLSS processing cycle.

d. Software Organization Considerations

The subroutine "CASER" currently controls (see Figure II-9) all double-line symbol generation. CASER generates two new symbols, left case and right case, until the current feature segment (i.e., data points) is exhausted. CASER then returns control to SIMBOL and MONITR for output processing of each of the new symbol segments by the appropriate output subroutine.

Concerning the above software organization, consideration was given to possible approaches to interrupting the process sequence for inclusion of "case intersection" software.

Facts to consider are:

- o Symbol segments produced by CASER are deposited into the common feature buffer for subsequent processing.
- o CASER returns control to SIMBOL/MONITR/OUTPUT for output processing.
- o Assuming all symbolization is completed just prior to output processing, very few current GLSS routines would be required for "intersection processing" and output processing.

e. Recommended Design Approach

The recommended design approach for the double-line problem is to store double-line symbols and associated "conflict" features on random storage for post-processing. This approach provides the best potential for generating clean intersections and is the least complex to implement within GLSS. This capability could be applied to any type of features requiring clean-up editing at intersections.

Basically the user would define a threshold value for edit

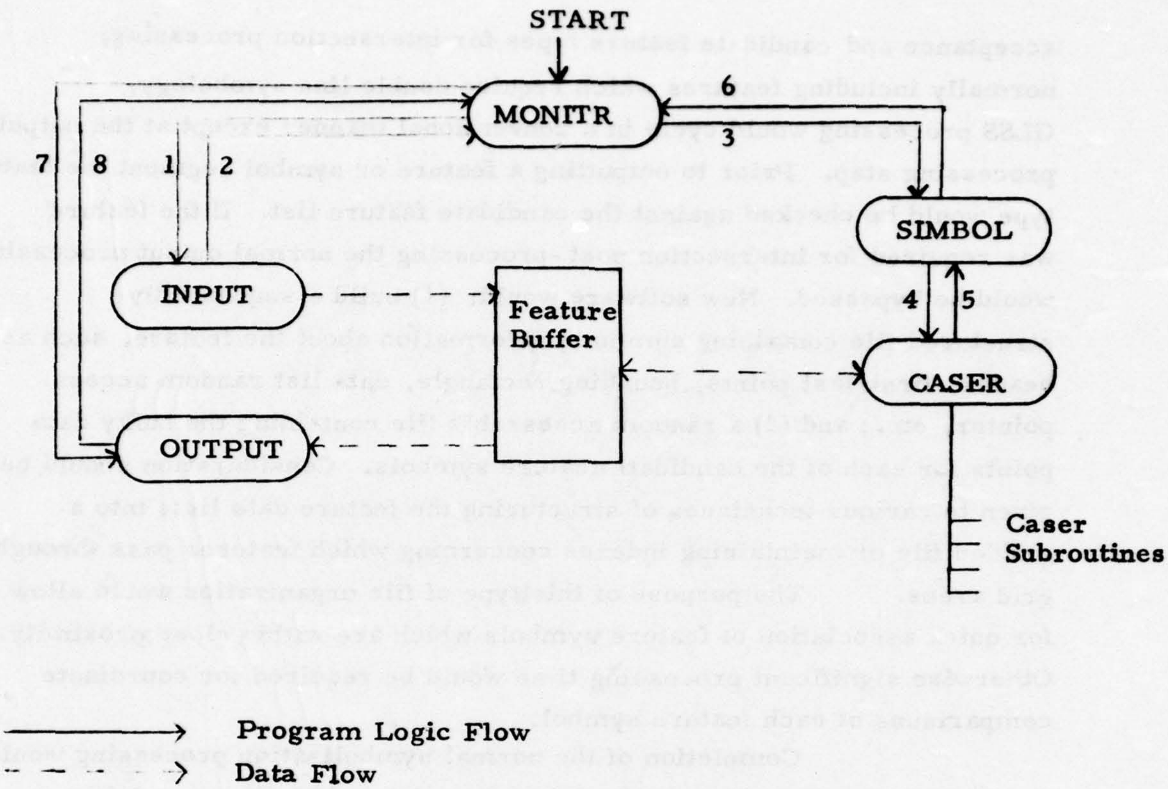


Figure II-9 General Flow of Case Processing

acceptance and candidate feature types for intersection processing, normally including features which require double line symbology. GLSS processing would cycle in a conventional manner except at the output processing step. Prior to outputting a feature or symbol segment the feature type would be checked against the candidate feature list. If the feature was required for intersection post-processing the normal output processing would be bypassed. New software would: (1) build a sequentially structured file containing summary information about the feature, such as, header, first/last points, bounding rectangle, data list random access pointer, etc.; and (2) a random accessable file containing the bulky data points for each of the candidate feature symbols. Consideration should be given to various techniques of structuring the feature data lists into a gridded file or maintaining indexes concerning which features pass through grid areas. The purpose of this type of file organization would allow for quick association of feature symbols which are within close proximity. Otherwise significant processing time would be required for coordinate comparisons of each feature symbol.

Completion of the normal symbolization processing would result in a tape file of symbolized features plus a disk file containing sequential and random sub-files of candidate feature symbols requiring intersection processing. The next step would be to overlay all symbolization application routines, and other routines not required for intersection edit processing. The overlaying software would be tasked with performing the intersection edit processes against the disk resident file. Following intersection editing control would be returned to MONITR and output processing would continue as normal.

2. System Improvement Considerations

The following is a list of potential system improvements and expansions for consideration by the Government.

- Alphanumeric Annotations - the capability to generate annotations for proof plotting and feature identification. Plotter alphanumeric subroutines are available and could be exploited by GLSS.

- Registration Points - GLSS processes and outputs only file information identified as features. Current users of GLSS can define registration points as features and thus such points will be plotted for registration. Consideration could be given to GLSS using control points on the LIS files for generating registration points.
- Color Separation Code Usage - GLSS generates a separation code on each symbol, based on the symbol specifications. This code could be used by plotter systems to select symbols for plotting on separation sheets or assigning different pen colors.
- Optimization of Memory Usage - GLSS currently requires approximately 37K to 41K words of memory (depends on the output module) for execution. Minor program overlaying strategies could reduce memory usage to 24K to 32K words.
- Additional Point Symbols - other geometric point symbols could be added to GLSS to provide a wider repertoire. Symbols to be considered includes: cave, building (rectangle), powerline pylons, rock and astronomic positions (plus symbol), etc.
- General System Usage - GLSS provides a generalized vehicle for input, data buffering, and output of feature files. New application routines, such as, line generalization, can be augmented for optional use during symbolization or standalone use for special applications.

SECTION III

OPERATING PROCEDURES

A. Operational Overview

GLSS is a batch system which operates on the UNIVAC-1108. Users of the system are responsible for defining input and output files, symbolization specifications and parameters to be applied, and special processing options; also the user is responsible for submittal of the job stream and necessary files, and review of outputs. The key to successful use of GLSS is understanding of the concepts and major elements of the system. Presented below is a discussion of the major elements of GLSS including:

- o Processing Capabilities
- o Operational Flow
- o System Resources
- o Control Cards
- o Symbol Specification Assignment
- o Job Reporting

1. Processing Capabilities

The following processing and symbolization capabilities are provided by GLSS.

- o Dashing - variable sizes of dashes and spaces
- o Casing - variable size casing
- o Ticking - full tick, half-tick, alternating half-tick, double tick
- o Point Symbols:
  - Circle
  - Dot
  - Arrow
  - Half-Arrow
  - Cross
  - Square

- Triangle
- Pyramid
- Arc/Chord

o Multiple Symbol Combinations (e.g., mixing of lineal and point symbols)

o Assignment of specified line weight and color separation codes to generated symbols

o File Input and Output Processing:
- LIS Table File (Input)
- Xynetics (Output)
- GERBER 2032 (Output)
- RAPS/MMS-32 (Output)

o Line Cleaning/Smoothing:
- Resolution Maintenance
- Line "back-up" edit
- Angle Bisecting
- Data Culling (based on slope change factor)

o Symbol Specification Control
- Building and updating of multiple symbol specification files
- Override to standard symbol specifications (at job run time)

o Job process and diagnostic reporting

o File Summary Capability - provides printout of feature headers for verification of classification codes

2. Operational Flow

Presented in Figure III-1 is a flow diagram of operations performed for symbolization processing.

3. System Resources

The hardware resources required for operation of GLSS includes the following:

o UNIVAC-1108

o Permanent Disc Space

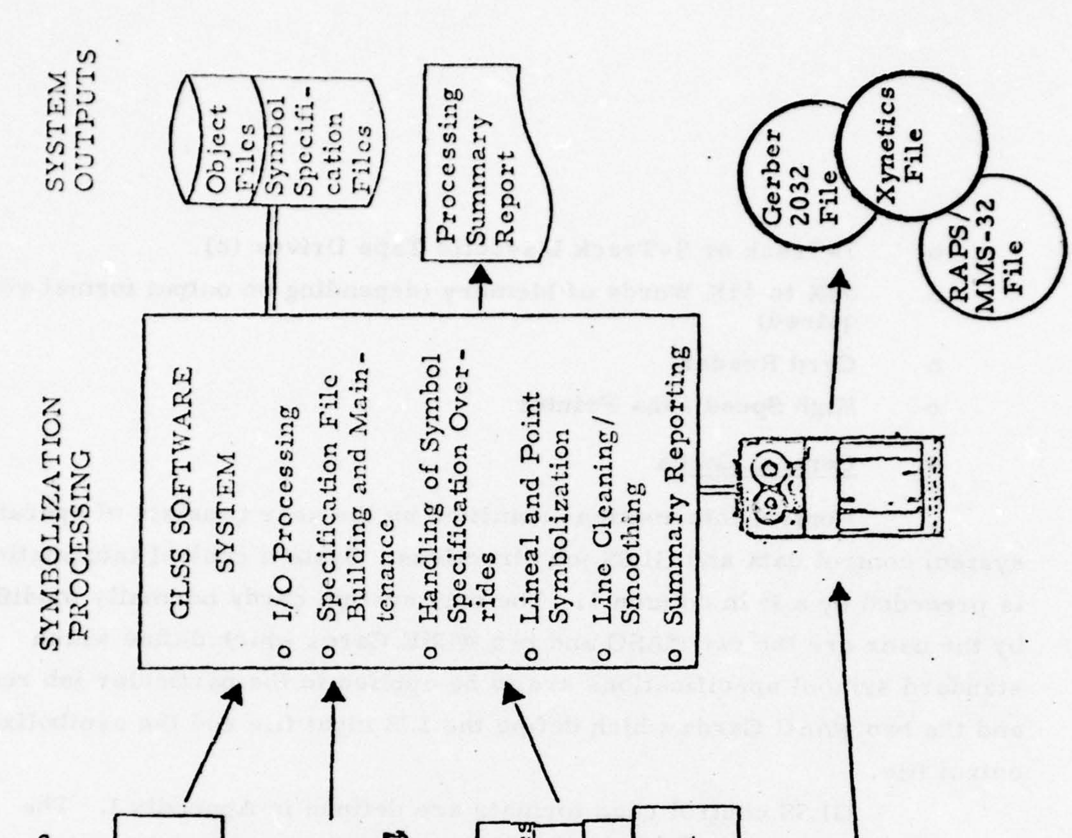


Figure III-1 - GLSS Operational Flow

- o 7-Track or 9-Track Magnetic Tape Drives (2)
- o 38K to 41K Words of Memory (depending on output format required)
- o Card Reader
- o High Speed Line Printer

4. Control Cards

Control information submitted by the user consists of operating system control data and GLSS job directives. System control information is preceded by a @ in column 1. The only system cards normally modified by the user are the two @ASG and two @USE Cards which define which standard symbol specifications are to be applied to the particular job run, and the two @ASG Cards which define the LIS input file and the symbolized output file.

GLSS control card formats are defined in Appendix I. The different control card types and their purpose are defined below.

INPUT FORMAT - defines the format of the input file.

OUTPUT FORMAT - defines the format of the output file.

SPECIAL DIRECTIVES OUTPUT - special option for generating a line center file containing symbol directives in the header records (not currently used at DMAAC).

SMOOTH OPTIONS - specifies the type of smoothing to be applied to the incoming features.

MINIMUM DISTANCE - defines the minimum distance between points or step size in inches to be accepted by SMOOTH optional processes.

MAXIMUM DISTANCE - defines the point spacing in inches which should be interpreted as trace vs. point-point types of data.

SLOPE DISTANCE VALUE - defines the distance in inches to be used for computing symbol orientations for lineal features or for SMOOTH Option 6 (Data Culling) the value defines the "inclination deviation factor" in degrees.

SPECIAL DIRECTIVES INPUT - indicates that symbol specifications are to be taken from the incoming header records instead of a standard specification file. (Not currently used at DMAAC.)

SPECIFICATIONS OVERRIDE - indicates the existence of symbol specifications which are included within the run deck and will be applied to matching features instead of standard specifications.

SYMBOL FEATURE CARD - defines the feature classification, descriptors, and color separation codes for the group of features to be symbolized with the override specifications.

SYMBOL SPECIFICATION CARD - specifies the attributes of a symbol piece.

END SYMBOL CARD - indicates the end of a specification override.

END CARD - indicates the end of GLSS control cards.

5. Symbol Specification Assignment

The system provides flexible and responsive techniques for defining symbology to be applied to feature groups. Since conventional products are symbolized based on standard specifications the system allows the user to build master specification files for each product. Product specification files are permanently stored and readily accessible at job run time. To allow the user to override standard symbol specifications for specific jobs the GLSS job stream allows the user to define feature symbology (maximum of 10 symbol overrides) which is to be applied to only that specific run. During job execution the override specifications are examined for feature matching prior to accessing a standard specification file.

Feature symbolization is defined in terms of symbol pieces. A feature can be symbolized by one or up to eight symbol pieces. Each symbol piece is described as to conformity to line orientation, symbol type, symbol size, and line weight (see Appendix I-12 for card formats). Figures II-4 through II-7 presents sample symbol specifications and resultant symbol patterns. Symbol pieces currently provided include the following:

LINEAL SYMBOLS	POINT SYMBOLS
LINE	DOT
DASH	CIRCLE
SPACE	TICK
CASE	HALF TICK
	ALTERNATING HALF TICK

POINT SYMBOLS (Continued)

ARROW

HALF ARROW

CROSS

SQUARE

TRIANGLE

PYRAMID

ARC/CHORD

B. Symbolization Operating Procedures

1. Job Planning

The key to successfully applying GLSS is comprehensive planning for the symbolization run. The user should be knowledgeable of the input file to be symbolized, the product being supported and therefore the symbology required, GLSS capabilities and constraints, and output format required. Each unique feature group contained on the input file which matches a feature group contained on the symbol specification file or override file will be symbolized and output, otherwise that feature group will default and be suppressed. If the user does not know exact header codes on the feature file, the input file can be processed by GLSSHEADSUM which will print out all headers on the file (see Appendix III). Any non-standard symbology required should be identified and verified for acceptability by GLSS. The quality of original line center plots and data point resolution should be examined to determine if some level of line cleaning or data reduction is desired.

The following smooth options are available:

0 - no cleaning is performed (original line center is passed to symbolization processes).

1 - deletes those data points (except special points) which are less than specified distance (MINIMUM DISTANCE) from adjacent points.

2 - Computes and passes the mid-points of all line segments (except those defined by special points)* and then performs option 1.

3 - performs option 1 and then checks and corrects for line back-up conditions for adjacent line segments.

4 - performs options 2 and 3.

5 - performs option 2 twice.

6 - performs option 1 and then data culling based on minimum angle specified by the slope parameter.

*Special points include end points, adjacent points which exceed maximum distance specified, and points which are flagged as special (e.g., intersections).

2. Job Set-Up

The user should obtain a copy of a standard GLSS job stream and modify the appropriate cards. The normal modifications are associated with I/O files and formats, specification files, SMOOTH options, and specification overrides. A sample job stream is presented below.

@RUN	(Estimate $\approx$ SUP Time should be 2 to 5. Sec. /1000 data points)
@HDG	****UNCLASSIFIED****
@ASG,A	DBM*UCPR-LT.
@ASG,A	DBM*UCPR-ATCS.,F/1/TRK/1/ (Spec. sequential disk file).
@USE	8, DBM*UCPR-ATCS.
@ASG,A	DBM*UCPR-ATCR.,F/1/TRK/5/ (Spec. random file).
@USE	2,DBM*UCPR-ATCR.
@ASG,T	3,U9H, NNNNNN (Input Tape Number)
@ASG,T	9,U9H, NNNNNN (Output Tape Number)
@XQT	DBM*UCPR-LT. GLSSXYN(GLSSGERBER or GLSSMMS)
1 INPUT	LIS
2 OUTPUT	XYNET (or GERBER, GERBPT, or MMS)
3 MMSDIR	NO
4 SMOOTH	$N (0 \le N \le 6)$
5 MINIMUM	.nnn (normally $.001 \le nnn \le .010$ in inches)
6 MAXIMUM	.nnn (normally $.010 \le n \le .100$ in inches)
7 SLOPE	.nnn (normally $\approx .040$ to $.080$ in inches or for SMOOTH 6 $000 \le nnn \le 090$ in degrees)
8 DIRECT	NO
9 SPEC	NO (or YES)

If card 9 contains a YES additional GLSS control cards are required (see Appendix I for specific card formats).

END

@FIN

Note: Underlined items above are optional job parameters.

3. <u>Execution</u>

Execution of a GLSS run, once properly set up merely requires submittal of the tapes for input and output (if tapes are not already contained in computer center's tape library) and submitting the job control deck.

4. <u>Job Run Validation</u>

Each job execution will produce a system job report and GLSS processing summary report. The system job report should be verified for normal termination. The user may also wish to note the CPU & SUP time for the execution. GLSS generates six reports which should be verified by the user.

Request Report - identifies the job options requested: input and output formats, smooth option and associated parameter settings.

Specifications Override Report - if overrides were defined by the user they will be listed for verification.

Input Report - provides a summary of feature information read from the input file: number of files, number of feature headers, number of data point records, and number of data points.

Symbolization Report - identifies the types of lineal symbolization performed and associated numbers of features processed, symbol segments and data points generated. If smoothing was requested the number of data points processed, deleted, and passed is presented.

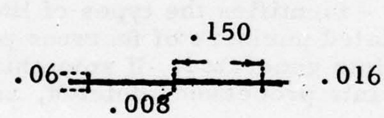
Output Report - provides a summary of feature information written on the output file: number of files, number of feature headers, number of data records, and number of data points. The number of feature headers and number of data records should be the same for both GERBER and Xynetics files.

Error Report - any error situations encountered by GLSS software processes are reported.

C. Specification File Building and Maintenance Procedures

1. Symbol Specification File Definition

The symbol specification directive generation is a major capability of the Graphic Line Symbolization System. The master specification file concept allows the user to define and build a set of symbol specifications for feature groups and selectively reuse the file for subsequent runs. The user must define: the file name, usually associated with a product (e.g., ATCS); all feature groups to be symbolized, and symbol specifications for the feature group. Any feature group not defined will be omitted from further processing. Product specifications normally contain symbolization patterns and drafting specifications to be applied to feature groups. The drafting specifications must be examined to determine if the patterns can be replicated by GLSS and the sequence of symbol pieces required. Every pattern must be uniquely defined in terms of conformity, symbol type, size, and line weight. For example the JOG specifications for drafting a Single Track Railroad (Feature 701) is the following:



GLSS specifications to generate the above symbols are defined as follows:

Conformity	Symbol Piece	Symbol Size	Line Weight
CON	LINE	.999	.016
CON	SPACE	.150	.000
CON	TICK	.060	.008

All symbol pieces required are conformal (CON) with the orientation of the feature line. The first symbol piece LINE tells the system to first generate the full line center (.999 implies infinity) on the output file and assign

a line weight of .016. The second symbol piece (SPACE) is then applied to the start of the feature for a distance of .150. A conformal tick of .060 is then generated at the current location and assigned a lin e weight of .008. The last two symbol pieces are iteratively applied to the end of feature line center. Examples of symbol specifications and resulting symbol patterns generated by GLSS are presented in Figures II-4 through II-7.

2. <u>Set-Up & Execution</u>

The symbol specification files are built and updated program (symbol name SPEC) that operates independently of GLSS. Two permanent data files are required for the execution of the SPEC. The first file (file code 08) is a permanent sequential data file which contains the feature descriptor codes and pointers to the associated specifications on the random file. The second file (file code 02) is a permanent random data file (35-word records) which consists of symbol piece directive specification data.

Symbol control data includes specifications for all symbol patterns to be applied to lineal features. Control data is input to the master symbolization specification files via data cards. The symbol specifications are organized by product and feature within product. A feature is defined in terms of: feature class, type, subtype, codified descriptors, symbol conformal/non-conformal information, color separation sheet numbers, symbol piece type(s), symbol piece size(s), and symbol line weight(s) for a specific feature group. Specification file BUILD/UPDATE control cards are presented in Appendix III. A sample SPEC run stream is presented in Appendix IV.

Execution of the symbol specification file build or update runs are accomplished after the user has generated the proper UNIVAC 1108 control cards along with his SPEC control directive data cards and symbol control data cards.

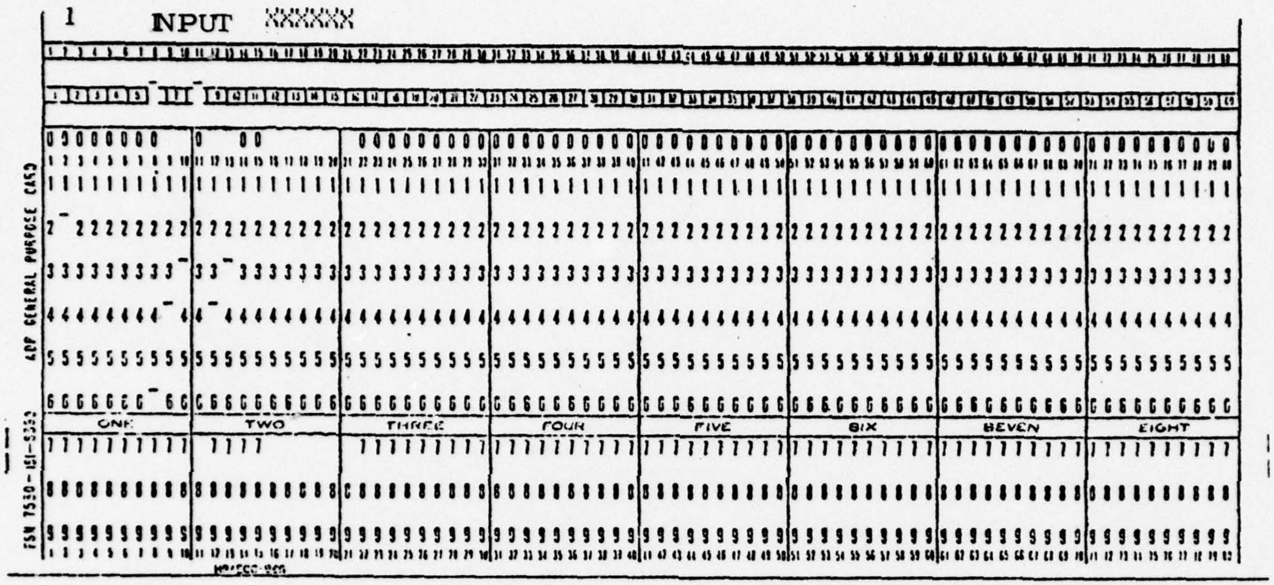
Validation of the job run is accomplished by examining the printer output report generated by the execution of the SPEC job stream. The printer output report will depict the mode of execution (BUILD or UPDATE). It will also depict the random record number built or updated, the feature descriptor definition codes, color separation sheet numbers, conformal/non-conformal information and the symbol piece type(s), size(s),

and line weight(s) associated with each feature descriptor defined. Also printed is the comment extracted from the symbol feature card.

APPENDIX I

GLSS CONTROL CARD FORMATS

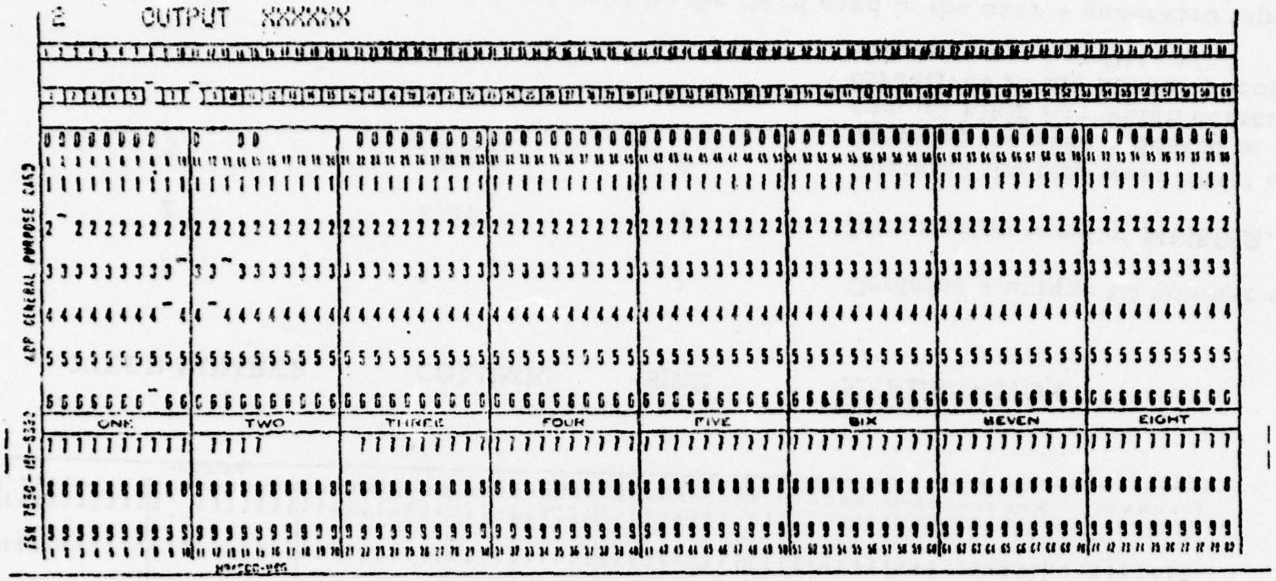
CARD 1 - INPUT FORMAT



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 1.
2	8-12	5	Description word of INPUT.
3	16-21	6	The input file format in which the symbolization will be input.

This control card is required and must be the first card in the user's generated input control cards.

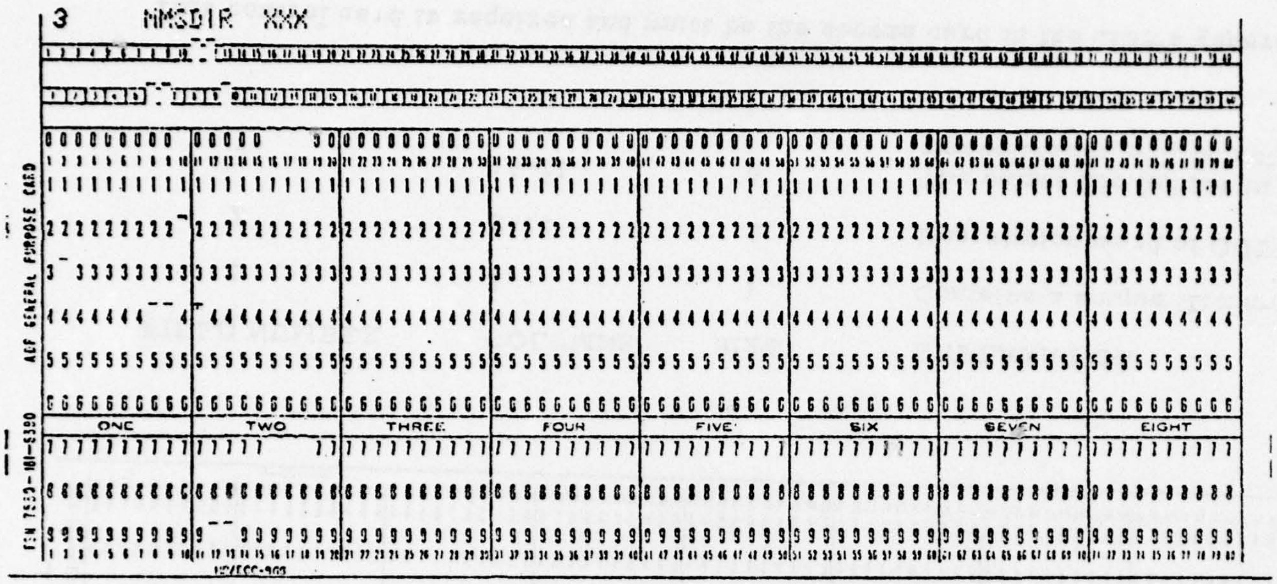
CARD 2 - OUTPUT FORMAT



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 2.
2	8-12	5	Description word of OUTPUT.
3	16-21	6	The output file format in which the symbolization will be output.

This control card is required and must be the second card in the user's generated input control cards.

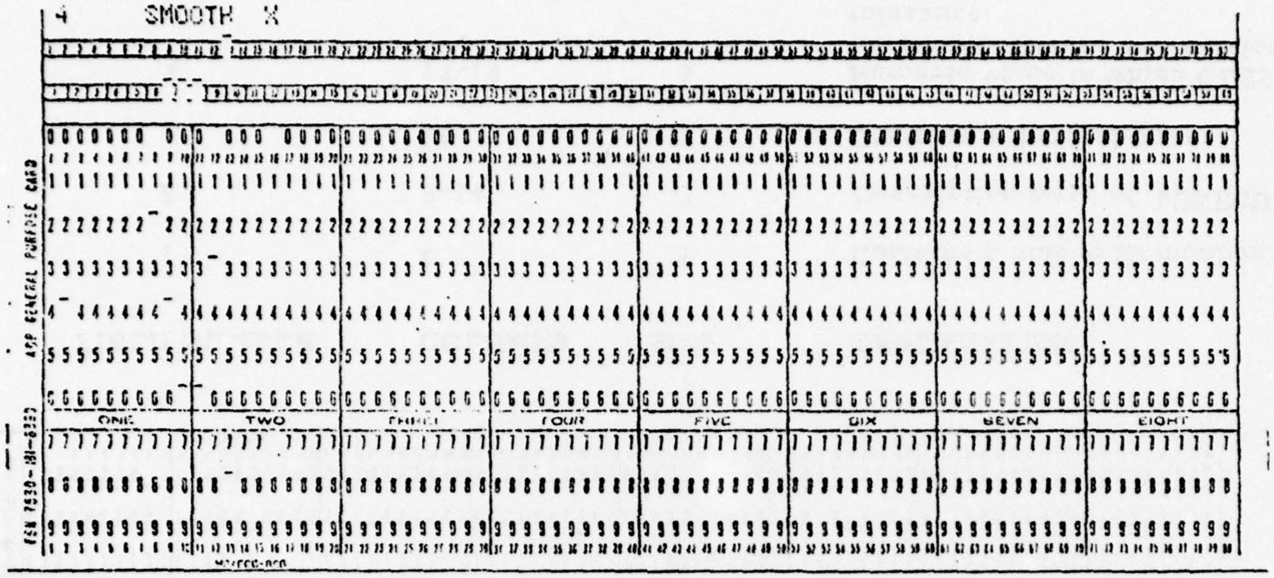
CARD 3 - SPECIAL DIRECTIVES OUTPUT



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 3.
2	8-13	6	Description word of MMSDIR.
3	16-18	3	Contains the text word "NO" or "YES" denoting the user's desire to write a special MMS file which contains symbol directives in the header record. (Not currently used at DMAAC.)

This control card is required and must be the third card in the user's generated input cards. If field 3 is "YES" the following control cards are not needed.

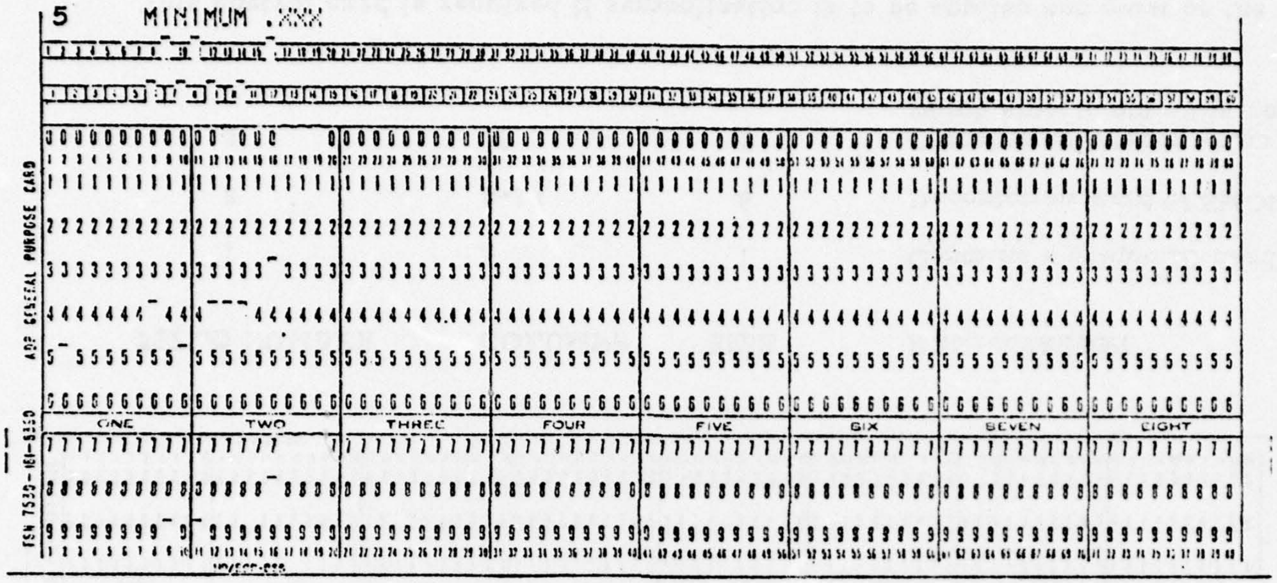
CARD 4 - SMOOTH OPTIONS



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID card number of 4.
2	8-13	6	Description word of SMOOTH.
3	16	1	Contains an integer value to tell the system which smooth option is requested by the user.

This control card is required if symbolization is to be applied and must be the fourth card in the user's generated input cards.

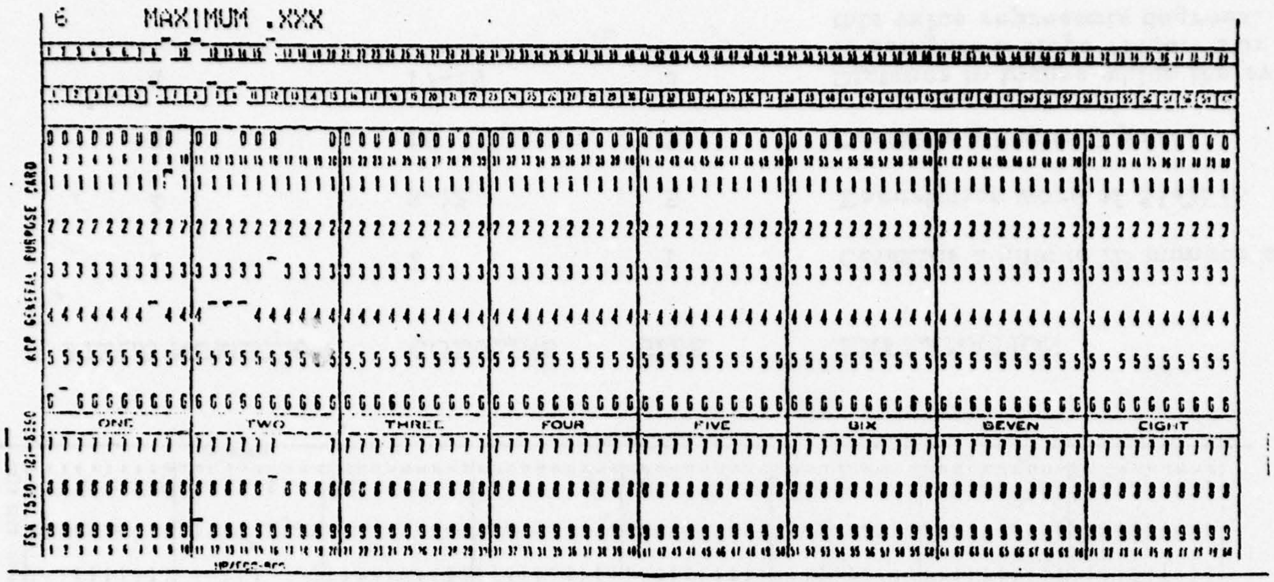
CARD 5 - MINIMUM DISTANCE



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 5.
2	8-14	7	Description word of MINIMUM.
3	16	1	Contains a decimal point.
4	17-19	3	Numeric value in which GLSS will use as a minimum distance and/or closest point tolerance.

This control card is required and must be the fifth card in the user's generated input cards.

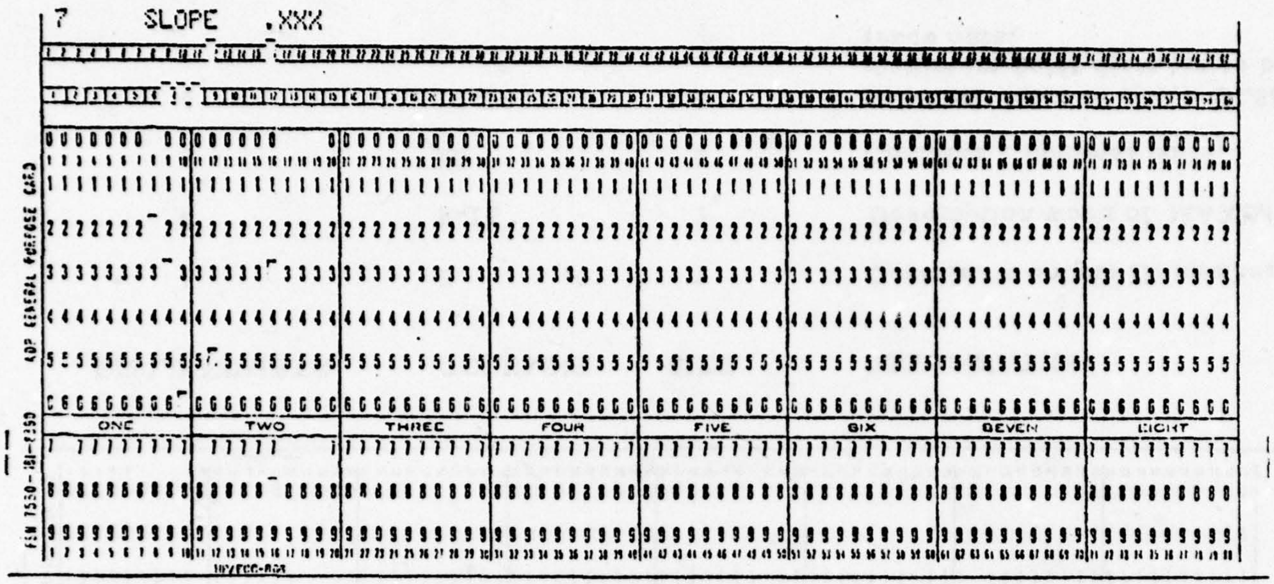
CARD 6-MAXIMUM DISTANCE



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 6.
2	8-14	7	Description word of MAXIMUM.
3	16	1	Contains a decimal point.
4	17-19	3	Numeric value in which GLSS will use as a maximum point distance to be considered as trace data.

This control card is required if symbolization is to take place and must be the sixth card in the user's generated input cards.

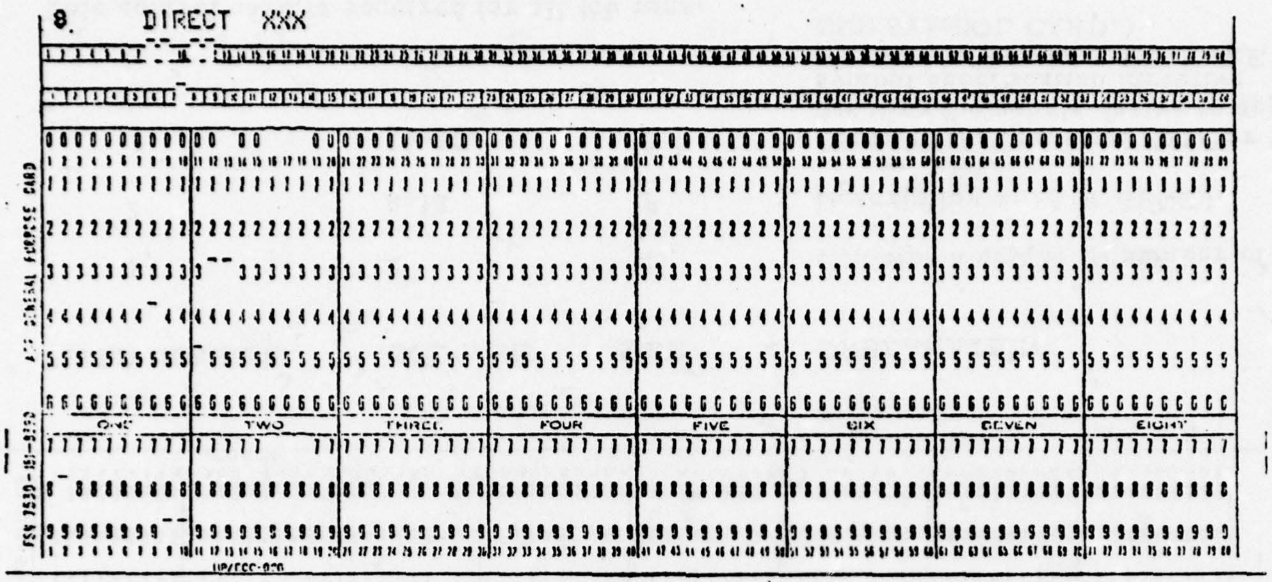
CARD 7 - SLOPE DISTANCE VALUE



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 7.
2	8-12	5	Description word of SLOPE.
3	16	1	Contains a decimal point.
4	17-19	3	Distance in inches which the system uses to compute a slope value. For Smooth Option 6 this value represents degrees.

This control card is required if symbolization is to be applied and must be the seventh card in the user's generated input cards.

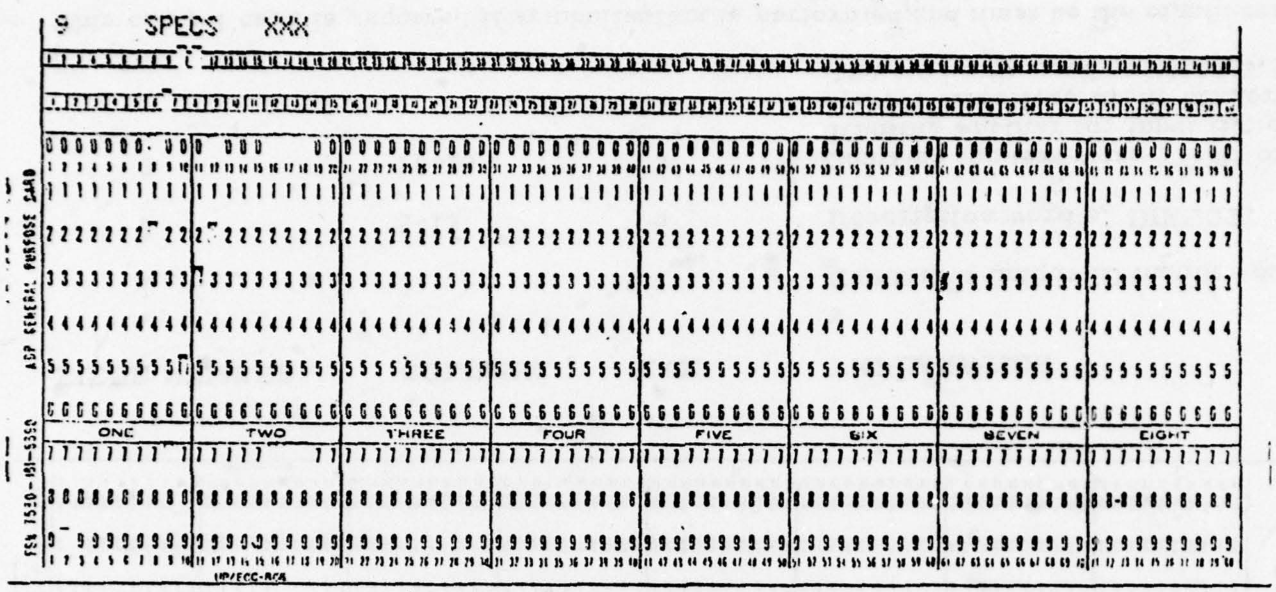
CARD 8 - SPECIAL DIRECTIVES INPUT



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 8.
2	8-13	6	Description word of DIRECT.
3	16-18	3	Contains the text word "YES" or "NO" denoting whether the input file contains the symbol directives in the header record. (Not currently used at DMAAC.)

This control card is required if symbolization is performed and must be the eighth card in the user's generated input cards.

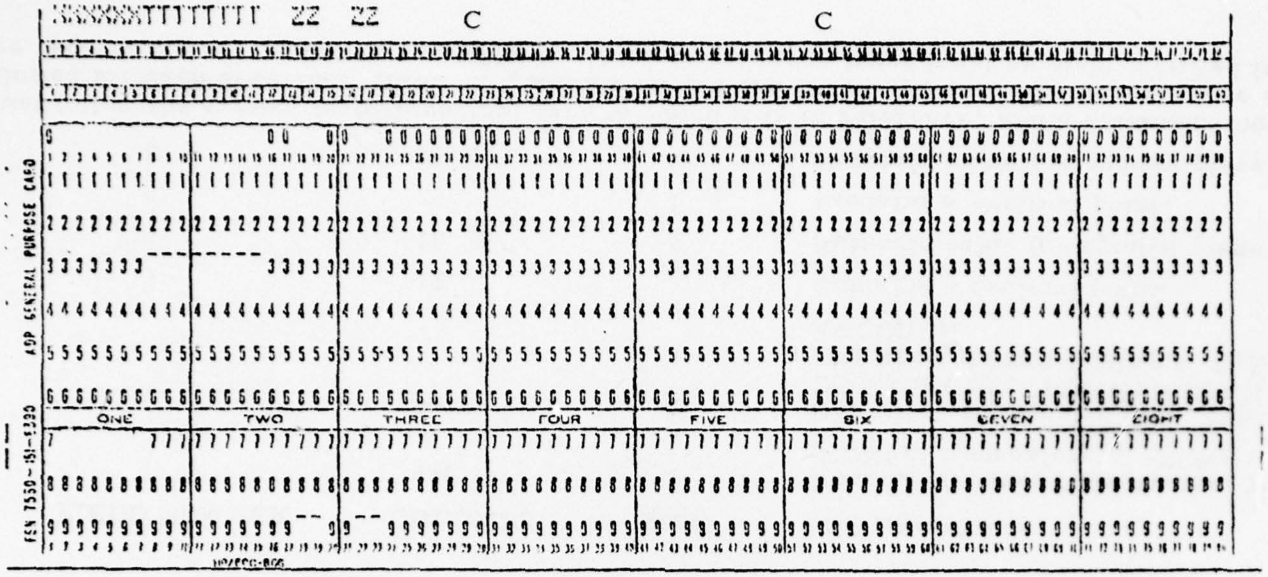
CARD 9 - SPECIFICATION OVERRIDE



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2	1	Contains a unique ID number of 9.
2	8-12	5	Description word of SPECS.
3	16-18	3	Contains the text word "NO" or "YES" denoting the user's desire to apply symbol specification directive overrides. (If YES see SYMBOL FEATURE CARD thru END SYMBOL CARD.)

This control card is required for all job runs.

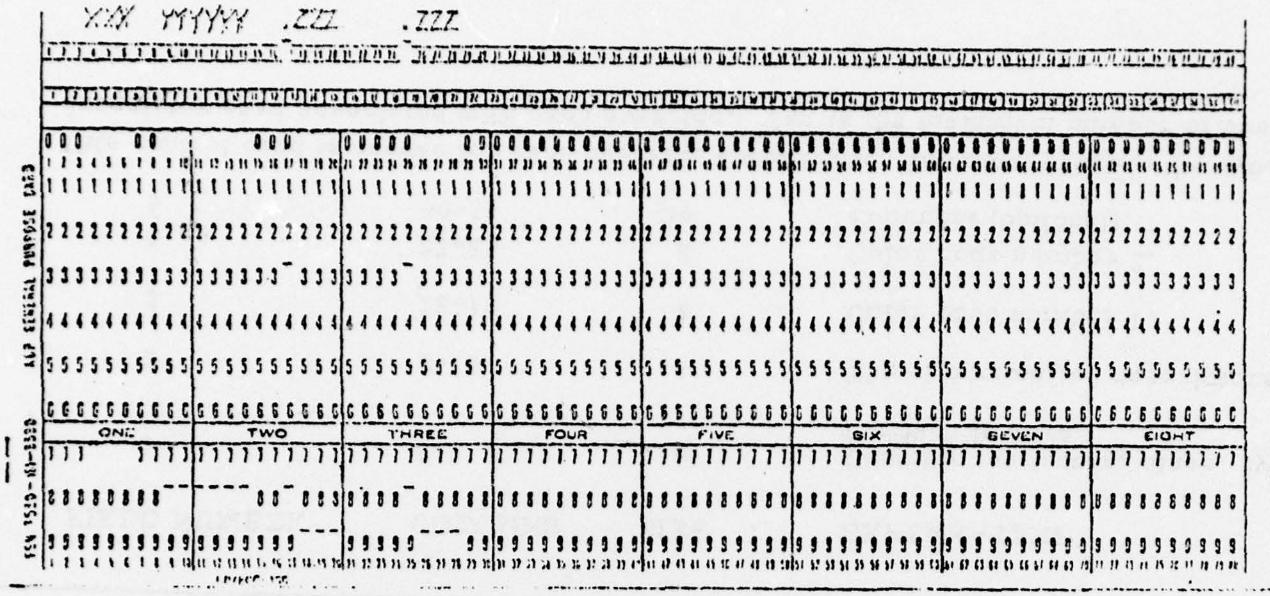
SYMBOL FEATURE CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2-7	6	Contains the feature class, type, and subtype for each feature.
2	8-15	8	The eight codified descriptors of a feature.
3	18-19	2	Color code number 1.
4	22-23	2	Color code number 2.
5	30-53	24	Comment (optional)

This control card is needed for each override specification and must precede the symbol piece descriptor card associated with each override. Ten is the maximum number of overrides.

SYMBOL PIECE SPECIFICATION CARD

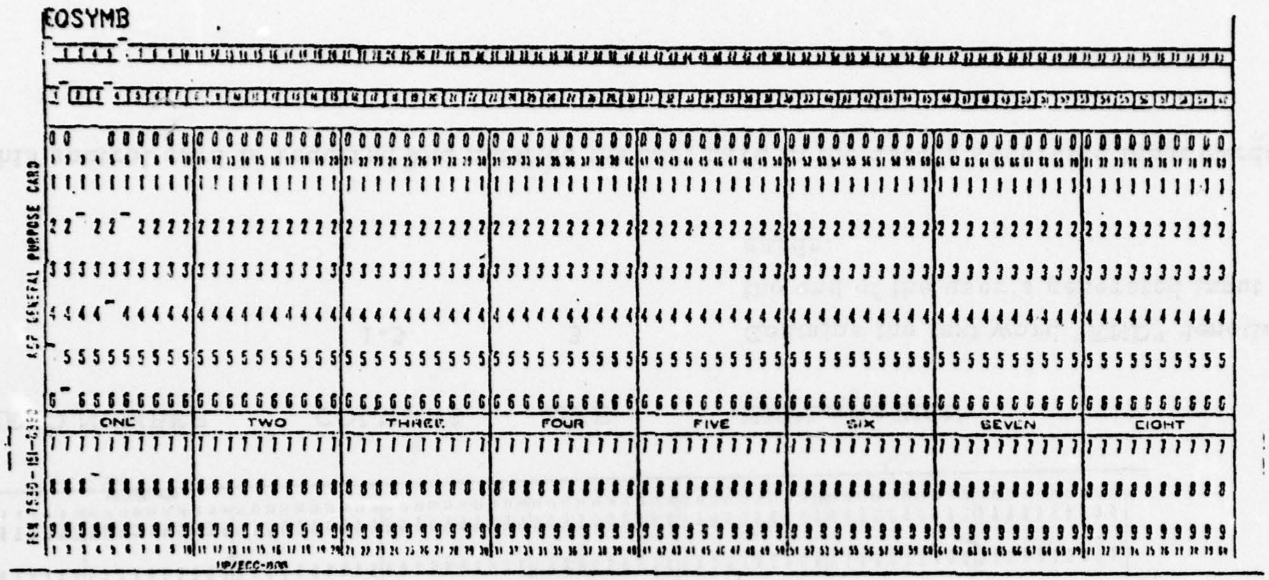


FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	4-6	3	Describes whether the symbol piece is conformal (CON) or nonconformal (NON).
2	9-14	6	Describes the symbol piece type (LINE, DASH, SPACE, DOT, CIRCLE, TICK, HTICK, AHTICK, CASE, ARROW, HARROW, CROSS, SQUARE, TRNGLE, PYRMID, or ARCORD).
3	17	1	Contains a decimal point.
4	18-20	3	Numeric value for symbol piece size. *
5	25	1	Contains a decimal point.
6	26-28	3	Numeric value for symbol piece line weight.

Must follow the symbol feature control card to which it is in reference, and a maximum number of eight is allowed for each override. Note: to generate an arc and chord symbol, two symbol piece specification cards are required; i.e., ARCORD and CROSS. (All symbol piece types must be right justified in columns 9-14).

* CIRCLE size should be ≤ .100"

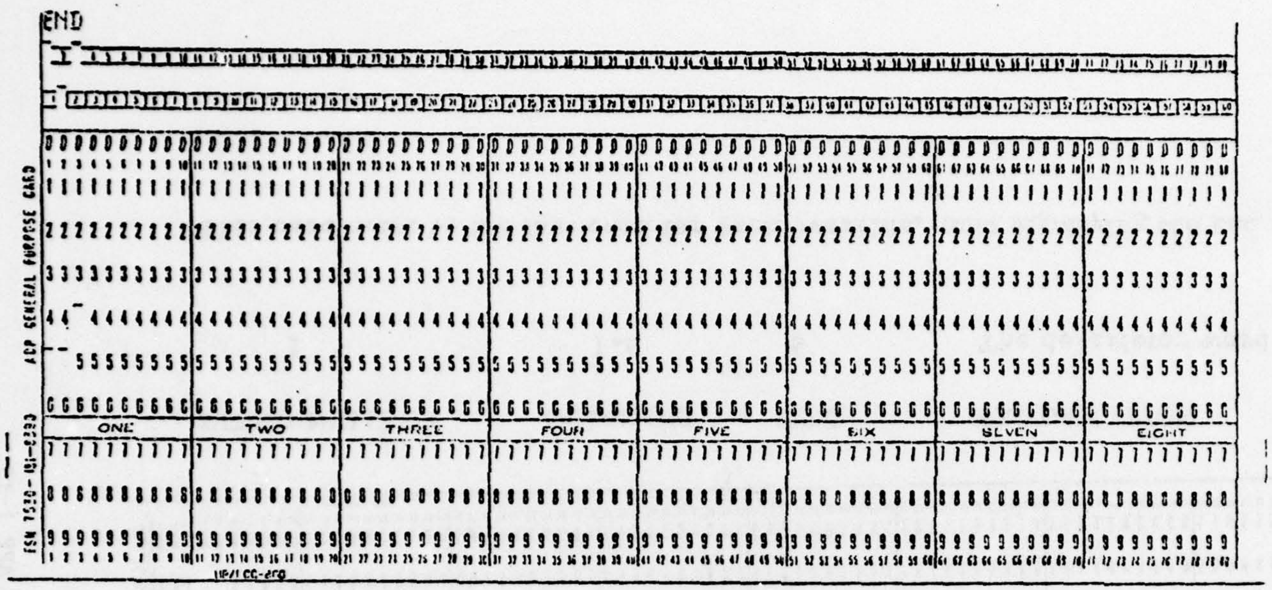
END SYMBOL CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	1-6	6	The descriptor word is EOSYMB.

This card must be the last card for each override, thus signifying the end of the override.

END CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	1-3	3	Contains the text word "END" denoting the end of the user's generated input cards.

This control card is required and must be the last card in the user's generated input cards.

APPENDIX II

GLSSHEADSUM JOB STREAM & OUTPUT FORMAT

```
@RUN
@HDG
@ASG,A    DBM*UCPR-LT.
@ASG,T    3,U9H,nnnnnn (LIS Table File Tape Number)
@XQT      DBM*UCPR-LT.GLSSHEADSUM
@FIN
```

Information printed out includes the following:

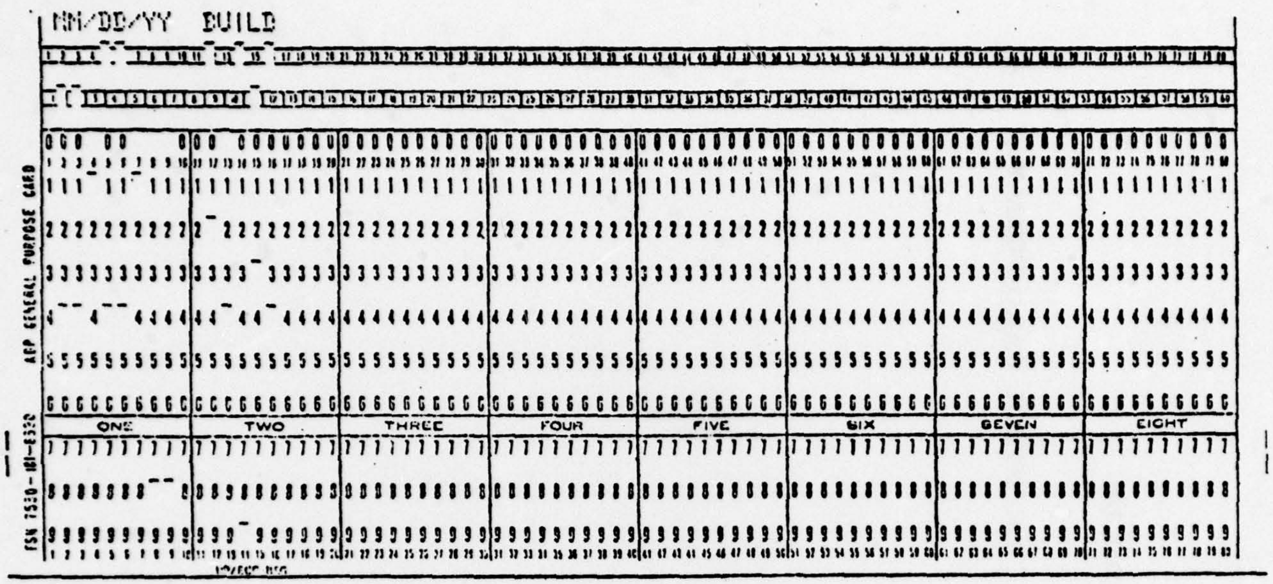
Feature Sequence #	Class, Type, Subtype	Descriptors
1	040809	61000000
	NUMBER POINTS =	222

•
•
•

APPENDIX III

SPECIFICATION FILE BUILD/UPDATE CONTROL CARD FORMATS

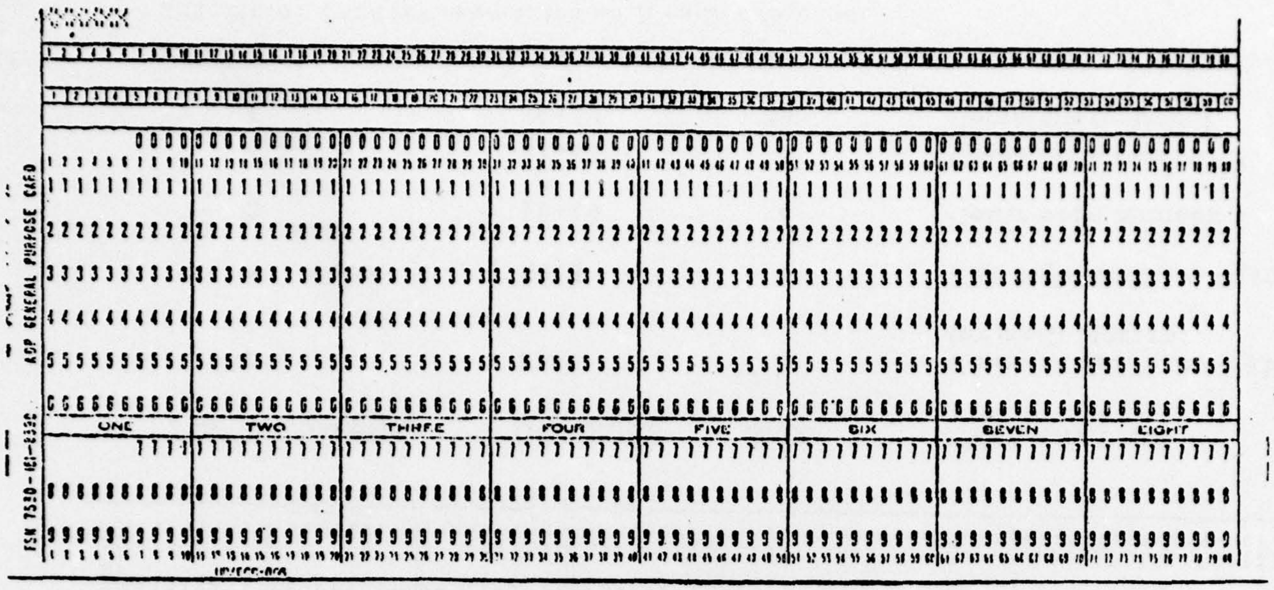
BUILD CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2-9	8	Contains the date of the run.
2	12-17	6	Mode of run:(BUILD) - build new specification file; (UPDATE) - update existing specification files.

This control card is necessary and must be the first control card. The first input control card for the symbol specification setup must contain the date and mode of run.

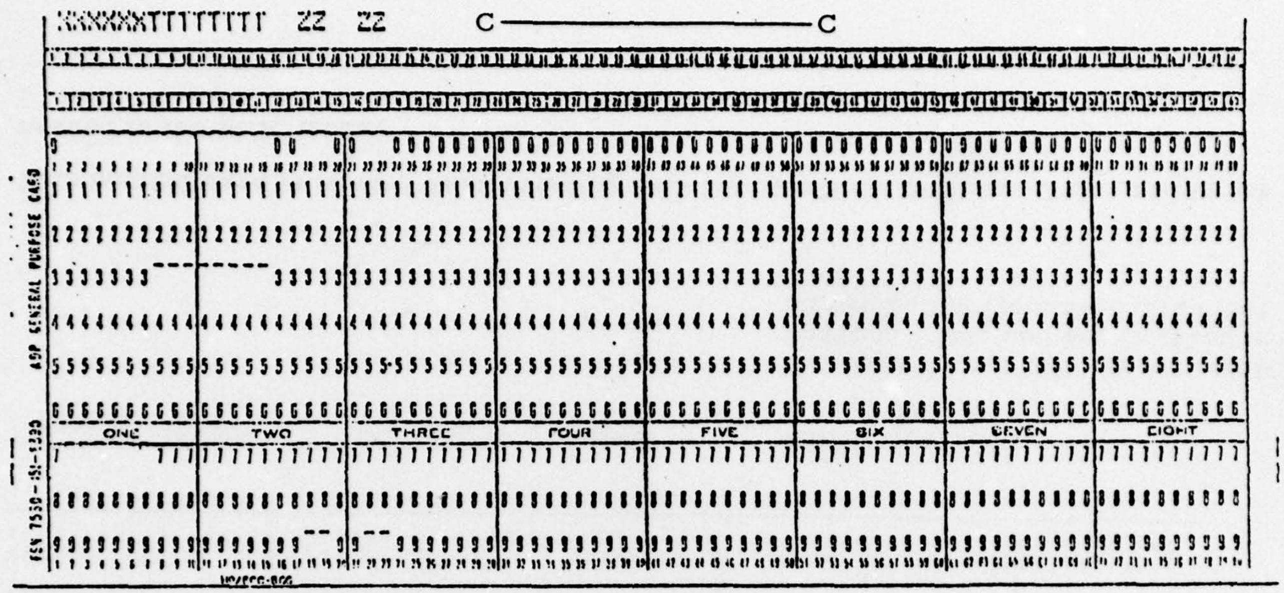
UPDATE RECORD CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	1-6	6	Contains the random record number to be updated. This number is taken from the output listing from the build mode.

This control card is needed when updating existing specification files. This card is not needed in the build mode.

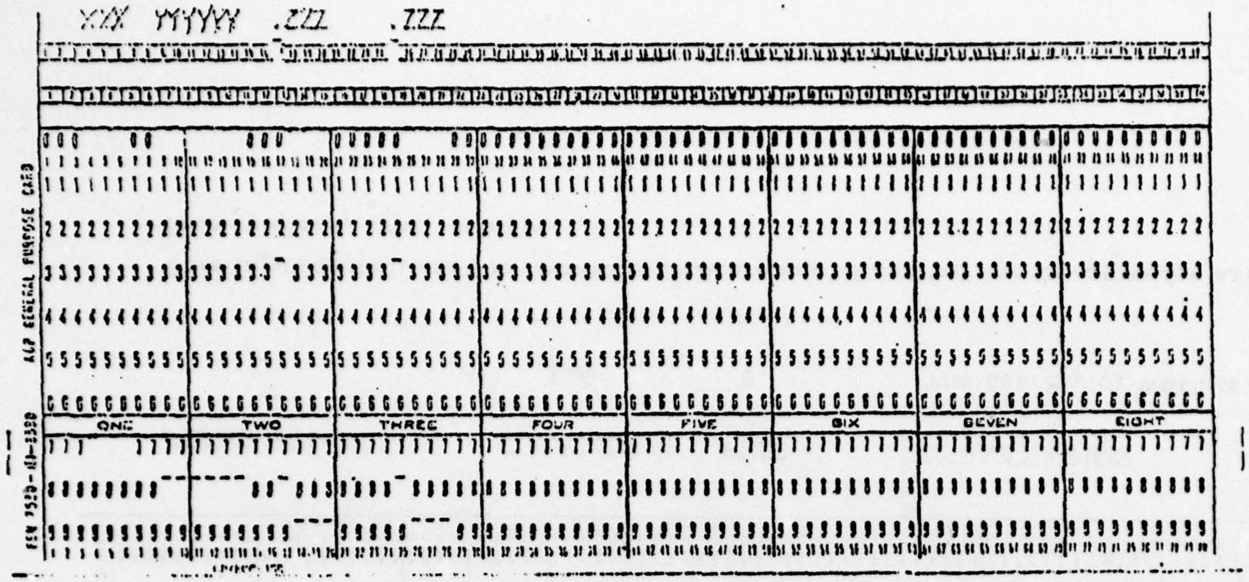
SYMBOL FEATURE CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	2-7	6	Contains the feature class, type, and subtype for each feature.
2	8-15	8	The eight codified descriptors of a feature.
3	18-19	2	Color code number 1.
4	22-23	2	Color code number 2.
5	30-53	24	Comment (optional)

This control card is needed for each symbol specification and must precede the symbol piece descriptor card(s) associated with each override.

SYMBOL PIECE SPECIFICATION CARD

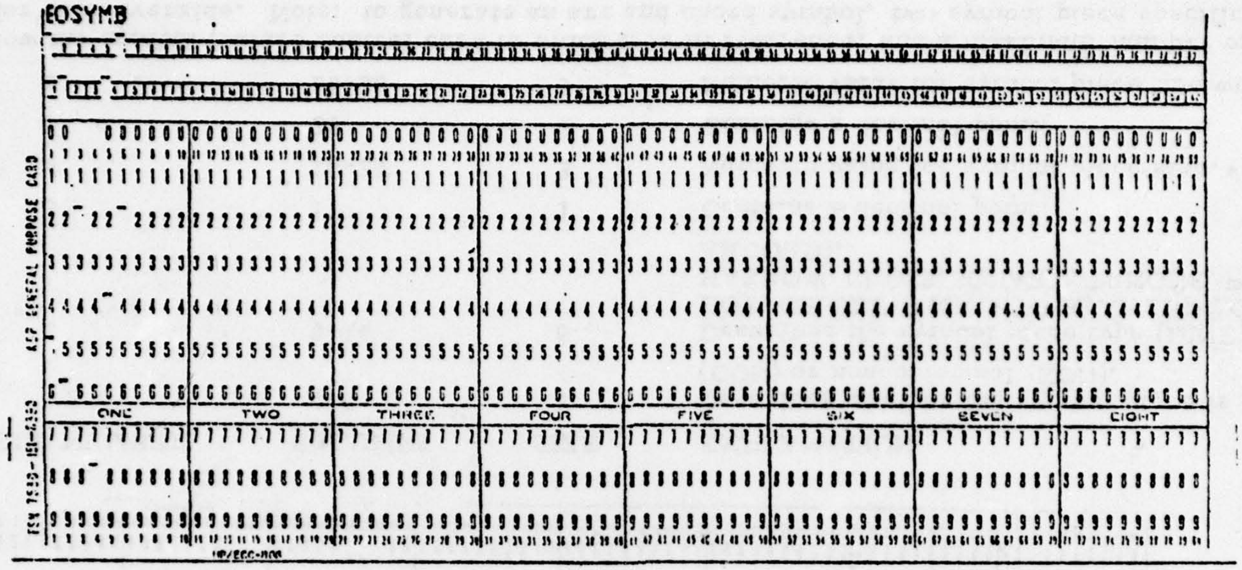


FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	4-6	3	Describes whether the symbol piece is conformal (CON) or nonconformal (NON).
2	9-14	6	Describes the symbol piece type (LINE, DASH, SPACE, DOT, CIRCLE, TICK, HTICK, AHTICK, CASE, ARROW, HARROW, CROSS, SQUARE, TRNGLE, PYRMID, or ARCORD).
3	17	1	Contains a decimal point.
4	18-20	3	Numeric value for symbol piece size.*
5	25	1	Contains a decimal point.
6	26-28	3	Numeric value for symbol piece line weight.

Must follow the symbol feature control card to which it is in reference, and a maximum number of eight is allowed for each override. Note: to generate an arc and chord symbol, two symbol piece specification cards are required; i.e., ARCORD and CROSS. (All symbol piece types must be right justified in columns 9-14).

*** CIRCLE size should be ≤ .100"**

END SYMBOL CARD



FIELD NUMBER	COLUMNS	SIZE	EXPLANATION
1	1-6	6	The descriptor word is EOSYMB.

This input control card is needed to indicate end of symbol specification for each feature class, type, subtype.

APPENDIX IV

SPECIFICATION FILE BUILD AND UPDATE JOB STREAMS

```
@RUN
@HDG
@ASG,A    DBM*UCPR-LT.
@USG,A    DBM*UCPR-ATCS.,F/1/TRK/1/    (File Name)
@USE      8,DBM*UCPR-ATCS.     (File Name)
@ASG,A    DBM*UCPR-ACTR.,F/1/TRK/5/    (File Name)
@USE      2,DBM*UCPR-ACTR      (File Name)
@XQT      DBM*UCPR-LT.SPEC

 11/12/75    BUILD
 01100000000000  01  01      CØMMENT FIELD
   NØN   SQUARE  .250      .008
   NØN    CRØSS  .100      .008
   NØN   ARCØRD  .100      .008
EØSYMB

.
.
.
@FIN
```

SAMPLE JOB STREAM FOR UPDATE MODE

```
@RUN
@HDG
@ASG,A     DBM*UCPR-LT.
@ASG,A     DBM*UCPR-ATCS.,F/1/TRK/1/          (File Name)
@USE       8,DBM*UCPR-ATCS.          (File Name)
@ASG,A     DBM*UCPR-ACTR.,F/1/TRK/5/          (File Name)
@USE       2,DBM*UCPR-ACTR           (File Name)
@XQT       DBM*UCPR-LT.SPEC

 11/12/75  UPDATE
    3
 05070000000000  03  04
   CØN    LINE  .999     .008
   CØN   SPACE  .250     .000
   CØN   HTICK  .060     .008
EØSYMB
   20
 31070000000000  01  01
   CØN    LINE  .999     .008
EØSYMB
.
.
.
@FIN
```

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.